卷首语

Foreword

够专业，就不怕失业

业界知名的Bob大叔在不久前发表了“软件技艺宣言”，向外界大声宣告：

我们不仅要提供可以工作的软件，更要提供技艺精良的软件。

我们不仅要响应变化，更要稳步增加价值。

我们不仅重视个体和交互，更重视建立专家群体。

我们不仅注重客户协作，更注重建立卓有成效的伙伴关系。

作为敏捷宣言的升级版，这个宣言实际上是在提醒我们：程序员，更应该具备专业精神。

专业，意味着专注、敬业。

专业，意味着不论一时一事之得失，力求一点一滴之积累。

要想成为专业的软件开发人员，经典算法、数据结构，这些扎实的基本功缺一不可；日常工作中，培养优良的编码风格，学习杰出的工程实践，阅读传世的大师之作，精通业务领域知识，这些都是必做的功课。

不仅要埋头读经典，还应该抬眼看世界。如今是互联网时代，开源项目、优秀代码比比皆是，唾手可得。项目不忙，闲下来了，少看几章小说，多了解了解地球另一方的杰出程序员们如何写程序，不无裨益。上网不方便？找本《程序员》看看也是好的……

总而言之，我们应该搞清楚自己的价值，打造身为一个技术人员的核心竞争力。

当然，即使无法强迫自己对工作精益求精，短期之内也没关系。咱们国家目前软件相关产业还不发达，产业结构与分工层次尚未清晰，混个几年也没啥大问题。不过等这一阶段过去了，到那时人到中年，上有老下有小，如果没有一技傍身，一旦真失业了，该怎么办？

截至目前，已有130多个来自中国地区的软件开发人员在“软件技艺宣言”上留下了自己的名字。你愿意做出这样的承诺么？如果可以，不妨去http://manifesto.softwarecraftsmanship.org/签个名，翻过头来，以专注、敬业的精神来面对自己的日常工作，从今天开始，从当下开始，一切都不算晚。

郑柯

Programmer 程序员 CONTENTS
技术改变世界 创新驱动中国 ● http://www.programmer.com.cn

技术 Technology

调试之剑 Debugging Sword

月度关注 Monthly Focus

BI让企业更“聪明”

“BI”这个词，有些人听过，有些人没有。面对BI，我们有欣喜，有苦恼。大家都寄希望于BI能让使企业更加“聪明”，但有时候却事与愿违。本期月度关注将带您走进BI的世界，帮您揭开诸多BI的疑团。

P110

产品推荐 Products Recommendation

图书 Books

评论 Comments

Contents

Programmer 程序员

2009年5月刊 总第199期

主管：中国社会科学院
主办：中国社会科学院文献信息中心
出版：《程序员》杂志社
网址： http://www.programmer.com.cn
国际刊号：ISSN 1672-3252
国内刊号：CN11-5038/G2
邮发代号：2-665
广告经营许可证号：京东工商广字0188号

总编：黄长著 Editor-in-chief：Huang Changzhu
社长/常务副总编：张悦校 President：Zhang Yuexiao
副社长：蒋 涛 Vice President：Jiang Tao
编委会：黄长著 张悦校 陈洋彬 蒋涛 曾登高 韩磊
Editorial Member：Huang Changzhu Zhang Yuexiao Chen Yangbin Jiang Tao
Zeng Denggao Han Lei

执行主编：孟迎霞 Executive Editor-in-chief：Meng Yingxia
编辑部主任：孟迎霞（兼） Director：Meng Yingxia
编辑部副主任：欧阳璟 Deputy Director：Ouyang Jing
责任编辑：郑柯 周至 李雨来 郭晓刚
Editors：Zheng Ke Zhou Zhi Li Yulai Guo Xiaogang
特邀编辑：方梁 高昂 常政 彭一凡 赵健平
Contributing Editors：Fang Liang Gao Aang Chang Zheng Peng Yifan Zhao Jianping
美术总监：张浩祥 Art Director：Zhang Haoxiang
美术编辑：吴志民 Art Editor：Wu Zhimin
Tel：010-64351458
Email：editor@csdn.net

发行部 Distribution Dept.010-64351431
Email：sales@csdn.net

广告总代理：北京创新乐知广告有限公司
Sole Advertising Agency：Bejing CSDN Co.,Ltd
Tel：010-64376055
Email：ad@csdn.net
Marketing Dept：010-51661202（ext 149）
Email：market@csdn.net

读者服务部
Readers service Dept.
网上订购：www.dearbook.com
读者信箱：reader@csdn.net
地址：北京市朝阳区酒仙桥路14号兆维工业园B区3楼2门1层
Address: B3-2-1F, Zhaowei Industry Park, No. 14 Jiuxianqiao Road,
Chaoyang Dist, Beijing
邮政编码：100016
电话：010-64351425
传真：010-64348545

法律顾问：北京市中润律师事务所 王杰
Law Consultant：Beijng Hengsheng Lawyer Firm
印刷：北京盛通印刷股份有限公司
Print: Beijing Shengtong Printing Co., Ltd.
出版日期：每月1日
Publication Date：the first day per month
零售价：RMB 10.00元 新台币 260元 HK $ 25.00（港、澳）
US $ 6.00（海外）
Retail Price：RMB 10, NT$260, HK $ 25.00,US $ 6.00

巾帼不让须眉，改变计算机世界的女人

——2008年度图灵奖获得者Barbara Liskov

■ 文 / 吕娜

她是美国第一位计算机科学女博士，是第二位获得图灵奖的女科学家；每一种主流汇编语言都受其深远影响。她，就是芭芭拉·利斯科夫 (Barbara Liskov)。

剑桥大学计算机女科学家Karen Sparck Jones的名言广为流传："计算机是如此重要，因此不能把它只留给男人去做！"计算机世界如此多娇，引无数女英雄竞折腰。有"计算机界诺贝尔奖"之称的图灵奖曾由男性垄断40年，今年6月，2008年度图灵奖将第二次授予女性科学家——Barbara Liskov，以表彰她对编程语言和系统设计方面所做出的实践与理论基础，尤其是数据抽象、容错和分布式计算方面的贡献。

Barbara Liskov，本名Barbara Jane Huberman，1939年生于加利福尼亚，是家中四个孩子的长女。1961年在加州大学伯克利分校获得数学学士学位，作为班上唯一的两名女生之一，她一直保持低调。

20世纪60年代，计算机科学成为一种职业，那时的科研环境对女性来说相当寒冷。作为一个在"男人"领域工作的妇女，Liskov经历了许多尴尬。本科毕业后她申请普林斯顿大学的研究生，但获得冰冷的回复：普林斯顿不接纳女性，无论是否具有本科或研究生水平。不久后，她在Mitre公司找到一份工作，同期入职的男同事没有更好的资格，却具有更好的职位和工资。Liskov无视这种公开对妇女的歧视，她有一种安静的自信，没有她不能克服的障碍。她认为，"发生的不公平的事情，并不与我直接相关，我想也许正是这种态度，使我已经适应这些年来此类处境。"

1968年Liskov在斯坦福大学计算机科学系获得博士学位，导师为1971年图灵奖得主约翰·麦卡锡。Liskov的博士论文题目是国际象棋残局程序，这篇论文是一个里程碑，至今仍在被引用。可以说，她是历史上为数不多的与程序设计直接相关，或者说程序员出身的图灵奖得主。

经过多年的努力，Liskov成为男性主导的计算机科学世界里最鲜艳的一抹亮色。在鲜有女性担任教师的1972年，她击败了一些强有力的男性竞争者，成为麻省理工学院的第二位女教师（计算机系的第一位）。2001至2004年，Liskov在麻省理工担任了3年计算机系副主任，这是第一位担任此职位的女性。但与之相比，她更热衷于研究工作，在过去的30多年中，她的研究成果给整个计算机科学带来深远的影响。

Liskov最重要的科研成果是为推动数据抽象使用所做的贡献，她在此领域的创新使得软件更易于编写、修改和维护，极大地提高了计算机软件的可靠性、安全性和易用性。Liskov领导了许多的重要项目，包括设计与实现了小型、低成本、交互式的分时操作系统Venus，面向对象数据库系统Thor，还有最近的Byzantine分布式容错系统。从实际项目中提炼出来的数据抽象思想，已经成为软件工程的重要精髓。

20世纪70年代早期，Liskov发明了2种计算机语言：CLU（一种支持数据抽象的面向对象编程语言）和Argus（一种分布式程序实现的高级语言）。这些研究成果成为现代编程语言的基础，支撑起整个现代应用软件行业，每一种主流汇编语言都受到其深远的影响，如C++、Java、Python、Ruby、C#。她与亚裔女科学家周以真一起提出的Liskov替代原则，是程序设计中另一个广泛应用的成就。这个原则已成为面向对象最重要的原则之一。

Liskov现任麻省理工学院电子电气与计算机科学系教授，美国工程院院士，美国艺术与科学院院士，美国计算机协会（ACM）成员，三本书的作者，发表技术论文一百多篇。2004年Liskov还被授予约翰·冯诺依曼奖章，以表彰她在编程语言和系统设计方面的卓越贡献。麻省理工官员拉斐尔·雷夫表示："她的杰出成就已经跨越了学术界的高墙，改变世界每日的生活。你每次和朋友交换邮件，检查银行账户，或者是搜索Google，你都在利用Liskov的研究成果。" ■

寻找更精彩的工作方式

我们所生活的时代非比寻常。纵观历史，每一次新的进步和繁荣几乎都始于人和人之间更加紧密的协作和更先进生产方式地采用。

人类社会的协作历史久远。大约公元前3200年古代苏美人用楔形文字颂扬王者的荣耀。这大概是最初的有组织的文字协作。当然这种协作的目的是强化权威和统治。它与我们今天所定义的协作不尽相同。随着时间的推移，协作不断地演进。下一个突破是1440年马丁古腾堡的印刷机。活字的采用使得书籍得以大批地印刷。这是促成欧洲文艺复兴的一个重要推动力。知识的传播由于此后的19世纪时发明的照相术和电话而大大加快。

20世纪，IBM发明的计算机把人类带入了数字化时代。20世纪80年代，个人计算机作为生产力工具得到广泛采用，从而形成了以文本软件为主题的工作方式。在文本文件统治工作方式的时代，新的计算机用软件工具开始出现。这些工具可以用于文档资料管理，编纂索引，进行模式化和个人信息管理。随之而来的是图形，数据库和分类管理软件。为主流企业级所使用的电子信息传递手段始于1989年的莲花电子邮件。

它的规模化采用比环球网的发明早一年，比万维网浏览器早三年。在90年代，电子信函是业务往来的主要方式。而即时信息则从个人使用渗入商业领域，成为又一个主要的业务交往形式。

下一步由全球性广泛合作带来的变革将更加深刻。如果各国都能够尽量把重建经济的资金用于以科技为本的智能化设施或项目的建设，我们很有可能会见证一次经济复兴。IBM认为应当把我们赖以生存的世界建设得更加智能化。要使个人、企业、政府部门、人造设施和大自然和谐互动。每一次互动都能够带来新的机会和效益。随着智能化程度的加深，人类将面临更深刻的生活改变。

智能化指的是数字式的智能不仅嵌入独立的事物当中，而且联合作用于整个社会系统，从而对社会生活的诸多方面产生影响。比如交通、供电、作物种植以及人类如何协作完成这些社会活动。

认识到智能化社会的巨大潜能，我们必须着手重建21世纪的信息技术。

20世纪，人类通过工业化革命把生产生活提升到了新的高度和效率从而使其更加经济和灵活。

智能化以及积极应对经济变化对于降低风险和提高效益至关重要。建设智能化社会的一大挑战是了解并协调人们和社会系统及运作几大要素之间的关系。所有这些因素都将和谐运转。

关键之一是人与人之间的合作要进入智能和高效的方式。这种方式将使人类做到以往之所不能。各种智能工作法已经开始实行。比如波音公司采用特别的软件把各种不同的信息综合起来生成各种预案并以此应对实际会发生的问题。波音是美国联邦政府在航空工业的一家主要承包商。波音公司不久前展示了一项信息综合技术。该技术做到了在一分钟的时间内识别出暴风雨范围内所有可以使用的机场。而仅仅是在2005年，为配合墨西哥湾卡特里娜飓风灾后重建所做的机场就花费了三天的人工。

凯尔医疗保健集团为医疗单位提供的技术可以调用远程的医疗手段并马上用于正在进行的复杂手术，设想病人正在接受心脏手术。而此时主治医生却遇到了意想不到的问题。通过联合通信手段，监视器所运行的软件可以在即时信息窗口传输断层扫描及其他放射影像，同时接收有关专家的意见，为进行中的手术提供宝贵的现场指导意见。

移动办公和现场办公的需要推动了移动网络的发展。IBM的商业价值研究机构预测在未来的十年中，将会出现20亿移动网络用户，而且人们与网络的交互会产生巨大的变化。人们将把五花八门的手持设备当作他们的移动办公室。用iPhone、诺基亚、斯普林特、Verizon无线服务和三星手持电话收发电子信件将成为日常的生活内容。人们还可以使用黑莓查看仪表盘，进入博客和短信，还可以随时随地处理公务。一些行业巨头，例如东芝美洲医疗系统，美国大都会生命保险和约翰逊控制系统等公司都在使用黑莓和协作软件把销售、客户、行政、产品设计和制造等各个方面联系起来。

社会交往已经成为重要的商业工具。它反映了一个新的现实，即我们的个人生活与工作之间的界线正在变得越来越模糊。更多的情况下，两者是同时得到照顾的。仅以几例说明，美国的实践法律学院、遍布世界的麦克森医疗服务、以及家用品供应商高露洁均在公司内部建立了自己版本的Facebook或LinkedIn。目的是在公司内挖掘建立更深层的和有意义的合作关系。在业务需要的时候，这种关系就会把最好的专业资源给予充分的发挥。这也从实际上提升了企业运营过程中的联合智能水平。

诸如此类的技术将不断涌现，进而演变成各种社会交流所需要的形态。更明显的变化是走向多种功能和形式的融合，包括移动软件、社会联系用软件、联合软件以及商业信息软件。社会交流已经成为新产品和新技术中必备的功能。比如，目前已经在法律和销售等行业使用的软件中出现了特定的社会联系功能。

另一个大有潜力的联合计算形式来自于社会联系软件与云计算的融合。很多公司已经建立了通过云计算提供软件和其他技术资源的服务模式。有些则通过互联网提供数据库资源。例如Salesforce.com、瞻博网络公司以及Amazon.com。

现在完全可以设想在任何地点和情况下处理任何事情。无论是在办公室、汽车里还是家中，都可以进入相关网络进行比如投标、招工、订货，与客户“见面”、与业务伙伴洽谈、与同事探讨新产品和新的服务项目等活动。至于这些活动具体由哪个部门或哪家公司主持，在以社交网络为基础的合作关系中，已经不再重要。曾经需要在办公室里做的工作已经搬到了“云”里。运营成本大幅度下降，资源取之不尽，但是使用资源的人只需支付所使用的部分。更多更好的想法和主意得以采用。产出将大大高于投入。这些都是智能工作的例子。智能时代正在向我们走来。

IBM Lotus 软件总经理 Bob Picciano

理性并购 有机发展

近年来，企业间的并购在全球范围内的IT外包服务产业中屡见不鲜，而不少国际公司通过并购，形成规模效应降低成本不断壮大的成功案例，也引来一些中国企业的纷纷效仿。但是，骤然袭来的金融危机也令其中的不少跨国公司遭受重击。如何走通并购发展之路，在目前的环境下远离与并购和扩张如影随形的风险，成为摆在企业面前的一个重要问题。

与其通过收购快速提升业务规模和销售额，我们更愿意坚持“有机”发展。提升跨地区的业务交付能力，进入新的业务领域填补业务空白，增强在细分领域的技术专长和咨询能力，才是我们制定成长战略以及考虑并购时所应遵循的原则。

有机发展，挖掘自身潜能是捷径

即使在经济危机时期，客户依然需要寻求IT外包公司的帮助以降低成本，提升效率，只是他们在费用方面要求更高。

全球化一直是Cognizant的基因，因此我们展开收购的目标之一即是提升我们在全球范围内的跨地区业务交付能力。2007年，Cognizant就收购了总部位于荷兰的金融服务咨询公司Infopulse，提升在欧洲，特别是在Benelux（比利时、荷兰、卢森堡三国经济联盟）的业务递交能力和客户基础。

其次，踩准步点，进入新的业务领域，填补业务空白。为了进入我们看好的新兴市场，我们应该考虑收购在这一新兴市场有突出表现的公司，并将公司的全球化优势和业务经验引入新的业务中，以提升我们在新兴市场的业务水平。

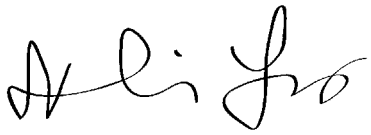
最后，提升公司专业技术能力。这是许多IT外包公司在进入新的服务领域时遇到的挑战，也是我们进行收购时考虑的因素之一。2007年，我们收购了MarketRX，增强了跨生命科学价值链的系列产品。同时收购还为公司带来了共由75家生命科学客户组成的大规模客户群，包括所有规模最大的20家制药公司和五大生物科技公司中的四家。

严格自律，恪守职业道德

在并购过程中，自律和职业道德往往容易被忽略。所以，我们需要谨慎把握，认真、审慎地履行自己在法律和道德上的义务。因为并购过程可能会对目标公司造成破坏，而市场上的不利传言则会使目标公司内部的员工和股东感到十分不安。所以，我们应该长期坚持的一项政策就是绝不公开论及具体的并购目标，无论这一公司是否在我们考虑并购的范围之内。这样做是为了顾全目标公司股东的感受和利益，是一种符合职业操守的行为。我们应该只有在公开宣布了并购计划之后才披露并购交易的具体内容。

提供更具成本优势的服务

面对宏观经济的不确定性，我们的客户都勒紧裤腰带过日子，所以我们的收购和扩张必须更加谨慎，并符合公司的战略目标。只有这样，我们才能保持和提升公司在现有和新兴市场的竞争力，合理处理业务交叉性战略配合，为客户提供更具成本优势的服务。



Cognizant 中国区总经理 杨洪林

迎接移动互联网应用的新纪元

很多人问我，2009年你最看好的技术和应用是什么？很简单，就是移动互联网应用。

伴随着信息化地不断深入，人们对信息技术发展的要求已经不再满足于“更强劲的性能、更低延迟的时间、更迅捷的处理速度”，取而代之的则是关于对时间及空间的考虑，也就是人们希望信息处理“无处不在，无时不有”。

而近年来，硬件技术的不断成熟、芯片技术的发展对于移动设备而言具有划时代的意义，它不仅具有更低的功耗，也能完全平滑移植基于传统PC的应用。尤其是3G网络，在取得高速网络功能上突破的同时也进一步普及了无线网络的应用。这些都为基于移动设备的互联网应用创造了条件，带来了机遇。

硬件更成熟了，网络速度更快了，用户普及程度更高了，移动互联网应用就形成了吗？这肯定是不可能的，因为缺少了最重要的一环：软件。

基于移动设备的互联网应用是一个三位一体的概念，“移动设备+软件+互联网”。作为用户，他拿着一个移动设备，这个移动设备能干什么他不知道，或者是他想做点什么但不知从何下手，这些问题都需要软件来引导、来实现，硬件只是“形体”支撑，而软件是“灵魂”。

如果非得在软件中寻找出一个大脑，那肯定就是操作系统，只有在这样的一个平台上，应用软件才能发挥功效，实现丰富多样的应用。红旗的Midinux就是一个为满足普遍需求而定制的操作系统，主要应用于MID（Mobile Internet Device）产品之上。让我们注意一组来自ABI Research的数据：在UMPC已经为MID铺平了道路的背景下，MID初始的平均售价应当在500美元左右，而到2012年将降至285美元左右；MID市场将从2008年的少量设备增长到2012年的9000万套。

这组数据不仅反应了当前的市场状况，更多的是对未来市场的预期。MID刚刚起步，还需要软件和硬件环境地不断完善和提升，以满足日益丰富的应用需求。值得高兴的是，过去一年里，MID产业链上的厂商都做了很多努力，如英特尔的凌动(Atom)处理器及Menlow，将自身强大的电脑芯片设计和制造能力向更小型化PC的方向转移；红旗Midinux2.0系统，为消费者提供更强的移动互联网体验，并联合OEM商实现未来大规模生产并降低价格。

硬件与软件全面得到提升，3G网络的启动是一股助推力，它为基于移动设备的互联网应用带来了一个加速度。3G网络的革新并不仅仅是我们打电话的时候能互相看到了对方，这只是3G带来的一个功能，或者是一个亮点，但这不是决定性的，毕竟100余年的电话史上虽然彼此看不见，可也不妨碍我们畅快通话。我认为3G网络真正带来革命性的东西就是基于3G网络的互联网应用，并且这些应用是在移动设备上实现的，高速率、低延迟的3G网络能带来更佳的用户体念，实现移动设备的“凌动”。

去年，被很多人称之为移动互联元年，今年则有了一个新名词：3G元年。明年呢？也许就是基于3G网络的移动互联元年了。

这是我们所能看到的趋势，也是我们不断努力并寻求突破的方向。

北京中科红旗软件技术有限公司
总裁兼CEO 贾栋

Oracle公司74亿美元收购Sun公司

4月20日晚间，Oracle宣布，将以每股9.50美元，总计74亿美元现金的价格收购Sun公司。Oracle表示，扣除Sun公司的现金与债务后，此项交易价值56亿美元。并且交易完成后，第一年内将使其调整后每股盈利增加至少15美分。

IBM此前曾报价每股9.40美元收购Sun公司，但本月早些时候，在Sun公司取消IBM的独家谈判权之后，双方之间的交易谈判破裂。

此项交易已获得Sun公司董事会的批准。Oracle预计交易将于今年夏天完成。

当事方

Oracle

Oracle的CEO拉里•埃里森(Larry Ellison)表示，Sun与Oracle的交易将会改变IT产业。Oracle将会成为唯一一家能够拥有从软件到磁盘的完整产业链的企业。所有的服务都完美地整合在一起，这样用户自己就不需要再去做些什么了。我们的用户能够因为系统的完整性而减少花费，同时在系统执行、可靠性和安全性上得到提升。

此外，Oracle对Sun的收购意味着Oracle将接管基于Solaris Unix的操作系统，以及两款主要的开源产品：Java程序语言和MySQL数据库。在Oracle的公告中指出，Solaris是市场现有的最好的Unix技术，并且承诺将对操作系统数据库进行优化。

Sun

Sun公司CEO乔纳森•施瓦茨在内部发送的邮件中提到："我不认为该声明发布之后就万事大吉。我相信，这是踏向新路途的第一步，这一步将把我们和我们的创新带向更广阔的市场，从而维持我们在这个世界上的角色。今天宣布的交易，经过后续的修订和股东批准，还需要几个月直至交易完成为止，我们仍然是独立运作的公司。无论这需要花多长时间，这个世界从今日开始改变了。

不过，重要的是要记住，并不是此次收购改变了世界，而是推动两家公司发展的人们改变了世界。我花了相当长的时间与Oralce会谈，我可以向你们保证，他们倾心关注我们的财务决算中未提及的因素——人。这是他们最高优先级的考虑。创建足够吸引人的环境，让我们最聪明的头脑能够继续创新和创造未来。"

企业声音

微软

微软CEO史蒂夫•鲍尔默："我也是刚刚听说，我需要好好想想，我对此非常惊讶。"

IBM

IBM全球副总裁、IBM软件集团

业界新闻

● 3月24日，统一通信解决方案提供商美国Aspect软件公司宣布，在国内市场推出"统一通信应用"软件与服务，将现有成熟的通信基础设施与最新的统一通信技术整合起来，改善整个业务流程的"效益之旅"。

● 3月31日，Novell在北京宣布推出专为新一代数据中心开发的操作系统——SUSE® Linux Enterprise 11。它是市场领先的关键任务Linux平台。利用这一平台，无论客户选择何种地点、何种方式，都能更加经济地部署工作负载，并得到Novell及其硬件提供商和独立软件厂商（ISV）全球合作伙伴的大力支持。

● SAP公司近日推出了致力于帮助企业优化绩效，降低IT成本的下一代软件套件——SAP Business Suite 7和用来满足不断增长的客户需求的创新举措，包括新的维护策略、新修订的SAP EnterpriseSupport服务功能。

● 3月24日，思杰系统公司在北京发布2009年中国战略，阐述在中国市场的发展规划。即推动虚拟化技术在本地的应用；深化合作，加深并拓宽产业生态系统；扩大对中国市场的投入，推动中国虚拟化市场健康发展。

● 3月26日，《中国社会科学综合地理信息服务平台系统》项目验收会举办。主办单位社科院协同中科院以及项目具体实施企业北京超图软件公司，共同完成了四个课题在该综合应用平台的汇报，这是GIS首次成功综合应用于国内社会科学领域。

● 3月27日，由普元软件主办的"SOA中国论坛"在北京召开。本次论坛的主题为"SOA从应用开始"。普元软件CEO沈惠中在大会中谈到，今天中国的企业已经开始实施SOA，并利用其架构技术和标准达到各种业务模块间和应用系统间的互联互通，从而最终实现SOA从应用级、部门级、企业级、生态级的渐进式发展路径。

● 3月27日，世界首个"中小企业全程电子商务服务联盟"发起仪式在北京召开。包括中国中小企业协会、中国工商银行、IBM公司、信用中国、通联支付、360安全卫士、图吧网络地图、富基标商企业供应链、AMT咨询集团等10大行业巨头与金蝶国际软件集团旗下友商网一起倡议发起服务联盟，宣布要联手解决中小企业管理提升、产业协作、商业贷款等三大难题。

● 3月30日，Oracle公司在北京举办"Oracle应用网格技术媒体交流会"。Oracle公司亚太及日本Oracle融合中间件

信息管理全球销售总经理Neil Isford表示，对IBM没有成功收购Sun，他一点也不感到遗憾，他认为IBM对Sun收购并不是很好的交易。

对于如何面对Oracle收购Sun之后的竞争，Neil Isford表示，IBM与Oracle还是有差异的，IBM一直强调开源的、灵活的系统，他认为收购Sun后，对于Oracle来说要面临战略选择，是延续原有专有的、封闭的道路还是走向开源的道路。

业界声音

OpenOffice.org团队

收购终结了对Sun未来不确定的担忧，对Sun多年来的帮助表示感谢。

Linux基金会

Linux基金会CEO Jim Zemlin表示，Oracle收购Sun将对Linux有利，因为Oracle在策略上和Linux为联盟，也是Linux发行商之一，以及主要的开源操作系统用户。

Oracle特别指出其对Java和Solaris的收购，并不意味着将减轻对Linux的支持，公司表示仍将对Linux一如既往地支持。就像IBM或惠普一样，继续对Unix投资的同时也保证Linux业务的持续，承认用户的自主选择，同时也确保在所选操作系统上运行软件的最优化。

MySQL

MySQL资深软件工程师Ryan Thiessen在博客上以震惊为题，对并购消息感到吃惊，表示去年Sun收购MySQL时他的态度是谨慎乐观，但今年Oracle收购Sun，他却感到失望和担心，他说他无法想象Oracle会视Java、MySQL或其它是能创造利润的产品。

Oak价值基金联合经理拉里·科茨(Larry Coats)

未来两年后，收购Sun将使Oracle运营利润增长15亿至20亿美元。Sun的问题主要源于管理层糟糕的决策，其业务本身并不存在重大问题。

媒体观点

《华尔街日报》

IBM周一发布财报，利润和营收双双下滑。但更加郁闷的是，Oracle与Sun达成了收购交易。这一打击要远远超过其财报影响。

路透社

Oracle宣布以74亿美元价格收购Sun公司，这是继IBM收购Sun谈判破裂之后的又一大惊人消息。

社区声音

- 上海师联网络科技发展有限公司项目经理汪申光：Sun与Oracle合并——MySQL将亡。
- CBSi产品经理李宁：Oracle和Sun真的来了，微软最该哭！
- 用友软件项目经理秦健：对IBM而言，煮熟的鸭子飞了。
- 上海浩为首席架构师喻桃阳：IT界的三巨头终于确立了——IBM、HP、Oracle。
- 新华信国际信息咨询（北京）有限公司项目经理李征：Java + Oracle，这是一个多么完美的组合啊！

产品管理副总裁邹晓兵指出，应用网格是基于动态资源的汇集和分享，并针对中间件架构的一种方法，可对资源进行最佳分配，以最低成本实现更高的效率、更灵活的扩展能力和更高的服务质量。

- 3月31日，英特尔公司在全球同步推出以英特尔至强5500系列处理器为首的17款企业级处理器。英特尔公司称，这些处理器是自约15年前英特尔面向服务器市场推出英特尔奔腾Pro处理器以来最具突破性意义的服务器处理器。
- 3月31日，阿里巴巴集团旗下子公司阿里软件在京召开战略新闻发布会。会上，阿里软件总经理王涛宣布将投入十亿元巨资，向中小企业推广管理软件，并承诺未来三年免费，目标在三年内使中小企业管理软件普及率从10%提升至40%。
- 4月3日，IBM发布了首个面向中国软件园的“软件园软件交付服务平台经营模式”。该平台可以使软件园在全园范围内搭建软件交付平台，供园区内的企业按需租用所需的开发工具，从而帮助他们以更低的启动资金、更便捷的途径获得IBM的软件开发支持。
- 4月15日，EMC公司全球同时公布了其专为支持虚拟数据中心而推出的创新高端存储架构——虚拟矩阵架构。EMC同时推出了第一款基于此种架构的存储系统Symmetrix V-Max。
- 4月21日，一年一度的“IBM信息随需应变（IOD）大会”在北京召开。会上，IBM信息管理软件对“智慧的地球”四大关键问题之一——“新锐洞察”发表了深入解读。同时，此次会议上还发布了其全新业务分析与优化（BAO）服务。

会议预告

首届亚太地区Android技术合作大会即将召开

中国Android协会将分别于北京(5月16日)和上海(5月17日)举办首届亚太地区Android技术合作大会。大会特别邀请众多亚太地区的Android业界专家齐聚一堂，以期共同促进各地区Android产业、技术及社区的经验交流、技术分享，创造潜在技术合作机会，开拓Android技术的应用商机。

2009第四届中国互联网站长年会即将召开

由康盛创想（Comsenz）主办的2009第四届中国互联网站长年会将于5月17日召开。这是该公司和著名的站长社区落伍者第四次联合主办这一站长领域的年度顶级盛会。

韩锷春
甲骨文大中华区
开发者计划高级经理

助.NET开发人员一臂之力

越来越多的.NET开发人员选择Oracle数据库进行应用开发，以满足最终用户对性能、扩展性和安全性等方面更高的要求。Oracle、Microsoft和第三方供应商都提供了针对 Oracle 数据库优化了的 .NET 数据提供程序，以期获得Oracle数据库预期的响应性能和高级数据库特性，其中，又以Oracle Data Provider for .NET（ODP.NET）相关组件和工具走在最前面。

ODP.NET提供标准的ADO.NET数据访问，同时也提供特定于Oracle 数据库的特性，包括完全支持用户定义类型、各种缓存、存储过程、XML DB、数据访问性能优化以及真正应用集群（RAC）连接池等。比如，对Oracle DB 11g中新的Securefiles特性的支持，使得应用对扫描文件、照片、字处理文档和电子表格等非结构化数据的处理更加高效和安全。

在满足最终用户要求的同时，.NET开发人员也会期望得到较高的开发效率，即：开发基于Oracle数据库的应用也可以像开发基于MS SQL Server数据库的应用一样方便。通过Oracle Developer Tools for Visual Studio（ODT）这个Visual Studio插件，开发人员可以非常方便地浏览和操纵Oracle数据库中的各种对象和数据，可以通过“拖放”数据库对象的方式自动地生成相应的代码、可以在VS环境中直接编辑和调试PL/SQL代码等。

就这样，ODP.NET和ODT等组件和工具帮助开发人员实现了.NET应用程序与Oracle数据库的最佳集成。

霍泰稳
InfoQ中文站
主编

世界从今日开始改变

Sun的官方网站上已经打上Oracle的Logo，其并购消息也已经登上各大网站的头条。有人说，在互联网时代，总不会缺少劲爆消息，这句话再次被验证。不过描述此次事件最为传神的还是Sun公司CEO乔纳森给内部员工宣布收购消息时说的一句话：这个世界从今日开始改变了。

开源是自由的，需要集体智慧的参与。Java，包括GlassFish、NetBeans和MySQL等技术/产品的成功，都离不开Sun对开发人员和开源 社区的重视。当“自由”的Sun/开源社区，遇上以“铁手腕”著称的Oracle时，未来会是怎样？开源社区的模式是否会被改写？让人难以预测。

不过有一点似乎是肯定的，那就是几个重量级选手已经做好准备——决战云计算。虽然在Sun被收购之后，鲜有Google对此事的反应，但可以想象的是他肯定加快了自己在云计算方面的布局，也许还加快了在操作系统方面的研发速度。IBM的蓝云和智慧地球也已经等待多时，而微软的鲍尔默在惊讶两个公司的并购速度后，也会再次掂量Windows Azure的分量是否足够对抗“Oracle+Sun”的冲击。不过更多人的反应可能是：喔，原来埃里森此前所说“云计算是扯淡”的话才是真的扯淡！

陶震
Sun 中国技术社区
网站主编

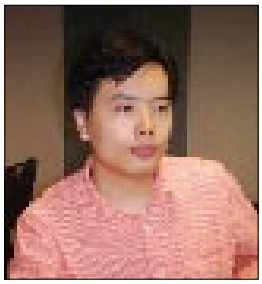
Project Darkstar——在线应用新天地

视频娱乐交互应用在工业国家中是仅次于电影的娱乐方式，它的流行为开发人员带来了巨大的机会。但是在创建和部署供数千人体验的在线娱乐交互应用时会带来技术和经济问题，并且，到现在为止，数十个平台还是相当庞大的系统，这不仅要具备多线程编程、分布式计算以及处理流程方面的专业知识，还要投入大量的资金维持后台资源。

Sun Labs Project Darkstar 提供了基于 Java 技术开源基础架构软件，该软件可以用于对于延迟要求较高的大型在线应用程序（如在线游戏等）。这不仅可以降低开发成本，而且还可以从根本上减轻甚至消除多人交互应用的开发人员在扩展、负载均衡、数据一致性以及故障恢复上所面临的问题，它现在受到该行业中众多领先企业的欢迎。

Project Darkstar 还提供了运行娱乐游戏的统一平台，从而使一个供应商就可以在同一个环境下为不同的人运行不同的娱乐游戏，就好像在一台应用服务器上。这样提高了部署过程中的资源利用率，最终还将减少成本。

微软技术



■ 主持人：Kaneboy
涂曙光，微软（中国）有限公司产品技术专家，博客堂成员。专注于.NET开发，从事Office System、SharePoint等产品相关的技术支持。

ASP.NET MVC 杀青

在经过了诸多Beta、RC之后，ASP.NET开发人员终于迎来了ASP.NET MVC正式的RTM版本。自ASP.NET诞生之日起，它所引入的WebForms的概念，就一直处于毁誉参半的状况。既有赞成者大声叫好，认为WebForms将Web开发带入了一个新境界，同时也有反对者质疑WebForms貌似简化了Web开发，实则让Web开发人员走上了一条“非典型”的Web开发之路。

不可否认的是，ASP.NET WebForms开发模型大大简化了Web应用系统开发，开发人员可以参照自己创建Windows应用程序中的经验，快速开发一个Web应用系统。但这样同时也“屏蔽”掉了ASP.NET底层的复杂性，让开发人员满足于“拖拖拽拽”的控件式开发，而不能基于一个良好的设计，来构建一个严谨而灵活的Web系统。很多大型Web应用系统的开发最终都陷入到众多功能和Bug的泥潭之中，其原因大多是因为系统的开发人员混淆了“快速构建”与“良好设计”之间的关系。笔者也时常可以看到某个“大型”Web应用系统，在一个页面的Code Behind代码文件中，充斥了无数的事件处理、业务逻辑，甚至数据库访问的代码。

这几年来，Web的地位不断地提高，企业中的业务系统也几乎都基于Web的方式来创建，ASP.NET开发人员自然希望能够有一个更严谨、更易规范化的框架。而近两年来，Ruby on Rails这样的框架开始大行其道，并受到Web开发人员的狂热追捧，无疑已经表明了众多Web开发人员的心声。ASP.NET MVC就是在这样的背景之下孕育而出的。它的出现并非为了取代WebForms，而是希望在WebForms之外，向Web开发人员提供另一种替代解决方案，一个能帮助开发人员更快更好地搭建复杂Web应用系统的框架。笔者从来都不认为WebForms就代表着“过时”、“不良设计”，实际上无数设计良好、功能丰富的Web系统同样是基于WebForms构建的。但ASP.NET MVC提供了一个更能“约束”开发人员的框架，更能“强迫”开发人员按照一种“更优架构”的方式思考。

www.asp.net/mvc网站上提供了ASP.NET MVC的下载和各种学习资料，ASP.NET MVC框架本身也以MS-PL协议公开了源代码。■

Java技术

■ 主持人：范凯
网络ID Robbin，JavaEye社区的创始人，开源软件的积极推动者和倡导人。

Google App Engine 支持 Java 引爆又一轮 Java 小高潮

在IBM和SUN之间关于收购的传闻满天飞的同时，Google App Engine宣布支持Java无疑是给Java社区打了一剂强心针。4月8日，GAE（Google App Engine）开发团队在官方博客上面宣布了这一消息，随后开放了一万名用户申请Java语言支持的名额。与此同时，在GAE的在线文档上面，已经发布了Java相关的开发者指导说明和文档，宣布GAE已经可以跑Java程序了。

GAE是Google公司云计算战略中的重要组成部分，它类似于一个虚拟主机的运行环境，用户可以把自己编写好的应用程序发布到GAE上，充分利用Google全球强大的计算能力、网络环境和软件支持，让自己的应用程序获得强劲的性能和超强的稳定性，同时还可以利用Google的分布式数据存储技术，轻易地开发出强大的云应用。

目前GAE对于Java的支持可以说相当的完善，除了少量的Java类库局限于GAE基础设施的架构无法提供之外，大部分流行的Java开源框架都可以得到完美的支持，另外GAE自身还提供了很多Web开发常用功能更好的封装，让开发人员可以更加轻松的在GAE上面开发云应用。像Spring，JPA这样流行框架或者标准都可以很好的运行在GAE上面，只是GAE不支持JDBC，这是因为GAE的底层是Google著名的非关系型分布式数据存储系统Big Table，而不是传统的关系数据库的缘故，但是由于GAE可以支持JPA，因此使用JPA也可以完美的绕过这个限制。

在GAE Java支持推出之后的短短几天，整个Java社区都在为之欢欣鼓舞，并在GAE上面尝试提供更多框架、语言的支持。Java平台的动态脚本语言Groovy很快推出了新的支持GAE的版本Groovy1.6.1，目前Grails还无法在GAE上面运行，预计将很快可以支持；JRuby团队也迅速发布了一篇介绍文章，指导JRuby用户如何在GAE上面运行JRuby，并通过JRuby成功的在GAE上面运行Ruby on rails；此外还有PHP的爱好者通过Java相关框架和类库的支持，也成功的在GAE上面运行了PHP程序；当然，作为Java平台的高并法动态语言Scala，也可以在GAE上面运行。

因此在GAE推出Java支持以来，除了C/C++之外，几乎大部分主流编程语言都可以很好的运行在GAE上面了。■

软件工程·项目管理

■ 主持人：潘加宇

UMLChina 首席专家，潜心研究和实践 UML/UP 相关技术的应用。

Barbara Liskov 获得图灵奖

ACM宣布把2008年度的图灵奖颁发给麻省理工学院教授Barbara Liskov，表彰她在编程语言和系统设计等方面的贡献。Liskov是第一位获得计算机科学博士学位的女性，但她被人所知往往是因为面向对象设计中一条很重要的原则：Liskov替换原则（LSP），大意是子类对象要能替换超类对象。LSP和上期说到的按契约设计有紧密的联系，用按契约设计来说LSP就是：（1）前置条件在子类中不能增强；（2）后置条件在子类中不能削弱。

Barbara Liskov的著作大多为科学论文，只写过三本书。唯一被译成中文的著作是一本教材“Program Development in Java: Abstraction, Specification, and Object-Oriented Design”，它的中译本名为《程序开发原理—抽象、规格与面向对象设计》，对Barbara Liskov的思想感兴趣者可以买来看看。

新世纪的图灵奖，已经有四位和面向对象技术有关联的科学家获奖了。他们分别是Ole-Johan Dahl和Kristen Nygaard（发明了最早的面向对象编程语言SIMULA-67）、Alan Kay（Smalltalk）以及Barbara Liskov。

NoMagic公司连续发布了其UML建模工具MagicDraw上的系列插件的更新。这些更新中有MagicRQ 16.0 SP2，NoMagic公司称其为“需求Hub”。通过MagicRQ，MagicDraw可以和各种流行的需求管理工具如Telelogic DOORS、IBM Rational RequisitePro无缝集成。MagicDraw的ParaMagic插件16.0版本也发布了，这个插件允许用户将Microsoft Excel、MATLAB/Simulink和Mathematica和MagicDraw的SysML模型集成。

Sparx Systems也发布了Enterprise Architect 7.5，一些新特性听起来相当诱人，例如序列图生成代码、状态图生成代码、活动图生成代码。另外，EA 7.5又分成了几种版本——业务&软件工程师版、系统工程师版、企业版和终极版。不管是MagicDraw还是EA，软件的个头都是越来越庞大。国内的UML工具厂商楚凡科技最近甚至发布了一个客户关系管理系统Trufun eCRM V5.0，看起来大家都喜欢往又大又多的方向走啊！■

Open Source

■ 主持人：叶亮

系统分析员，SCJP，Java 工程师，关注开源社区。

Sun is not set

过去的这个月中最热闹的事莫过于IBM收购Sun的传闻。最后的结果真是出人意料：北京时间4月20日晚，Oracle和Sun发表联合声明，Oracle收购Sun。与之前IBM收购传闻时的“对rumor不做评价”不同，Sun的CEO Jonathan Schwartz亲自发出E-mail，向全体员工确认此事；股价也应声而动。第二轮猜测和讨论开始：MySQL、Glassfish、NetBeans和Solaris的命运将如何？Oracle将怎样对待Sun的员工？收购后，Oracle是否将“垄断”数据库？IBM和HP会作何反应？无奈，跟大多数人一样，虽然可能会被影响，笔者唯一能做的就是每天看新闻。大家普遍认为，相比IBM，Oracle可能是更好的归宿。收购Sun后，Oracle摇身一变，软硬通吃；而其在Java和数据库方面的强大实力，更有能力延续Sun的技术。这样的格局应该会比IBM一家独大要好得多。

笔者联系了Sun在布拉格的工程师，问及收购对Sun产品可能的影响时，获得的“答案”非常正面：“Sun的很多产品已经有成熟的社区，不管Sun的命运如何，相信这些产品都能够在社区的支持下，健康的生存下去”。说的不错，理想的开源世界里，技术的兴衰不应该靠某个公司的一己之力，Linux是这样，Java也是这样。有网友戏言“为什么是Oracle来收购，不是联想呢？”Sun对开源做出了不小的贡献，根基也相当深厚，而这正是国内厂商严重缺乏的。希望以后这句话不再是玩笑。

Google刚刚发布了其App Engine对Java的支持。虽然EJB等大家伙还不能部署，但Java EE规范里面一些轻量级的组件都可以部署了，包括Servlet、JSP、JPA等；甚至还有Spring和部分Apache Commons组件；Hibernate支持的还不够好。所以可以把App Engine看作一个Web容器。拜Java所赐，JRuby、JRuby on Rails、Groovy、Grails还有Jython应用都可以跑了。这消息着实让Ruby社区兴奋了一阵。TheServerSide上有一篇对App Engine支持Java的评测，有兴趣的读者可以研究一下。■

移动开发技术

■ 主持人：马宁
微软最有价值专家，Windows Mobile 开发者。

野百合也有春天

经济危机的背景下，当大公司忙于收缩业务时，创新型的小公司却能够在某些新兴领域取得领先优势。Spb在CTIA 2009展会上发布了Spb TV，一款可以通过在线流媒体免费观看全球电视直播的软件。它提供了平滑的切换和频道预览界面，还提供基于Outlook的电视节目提醒。Twitter已经成为2009年最重要的媒体，Twikini是一款运行于Windows Mobile上的Twitter客户端软件，具有编辑、订阅、与媒体播放器整合等功能。

在CTIA展会上，Nevada发布了基于iPhone的Quickoffice for iPhone。支持在iPhone上编辑和管理Word和Excel文件。可见，Windows Mobile在移动办公领域的优势正在被这些第三方软件逐渐蚕食。当iPhone和Blackberry拥有了访问Exchange邮件的功能，Windows Mobile之前的后端优势将不复存在，这才是真正的危机。另一项针对LBS开发者的调查中，58%的非Android开发者计划将应用迁移到Android平台上，而向iPhone上迁移的意愿为40%，Blackberry为26%，Windows Mobile为20%。而Palm和Symbian的迁移意愿仅为8%和9%。这也能部分反映出开发者社区对移动操作系统的看法。

浏览器方面，基于WebKit的Windows Mobile浏览器Iris正式发布。由于采用了与Safari、Chrome相同的内核，网页加载速度比普通浏览器快25%，支持平滑的Zoom功能。Google也在积极促使iPhone和Android支持Gmail离线操作功能。基于W3C正在制订的HTML 5标准中，对数据缓存的支持。Gmail可在离线状态下进行读写邮件操作。

硬件设备方面，Acer推出了基于Nvidia Ion平台的紧凑PC。AspireRevo采用了Intel Atom CPU和Nvidia GeForce 9400 GPU，可以支持1080p高清播放和DirectX 10。

北京琦基推出了针对警察的AK 007手机，运行Windows Mobile，支持摄像头、GSM、WiFi和GPS，并采用了特殊加固外壳。美国Glacier推出了可以戴在手腕上的GPS设备，运行Windows CE，支持蓝牙、WiFi。这两款产品都是针对特定领域的移动设备，这也许就是野百合们绽放的未来。■

数据库技术

■ 主持人：王翔
软件架构师，主要研究方向为XML、.NET、领域设计和PKI应用。工作之余喜爱旅游、写作和烹饪。

拿些数据，放云上
取些数据，从云上

也许就像微软S+S战略后面极力希望将用户留在Windows平台的意图一样，Oracle也不愿意把大好的数据库市场从手中放弃掉，以至于拉里•埃尔森在宣称“到底什么是云计算呢？完全是胡扯”。诚然，现在的云产品（包括：微软、Google、亚马逊、IBM…）主要还停留在简单信息存储和访问的初级阶段，但从微软、IBM的下一代数据库产品中不难发现，基于云平台的关系信息体系已逐步形成规模。

本月，IBM联合SAP完成了一个基于云平台的“实时机动性应用”（Real-Time Application Mobility）项目。由于商业机会和市场的不确定性，很多企业的IT吞吐量井喷经常会发生在现有的数据中心之外，业务流程优先级及相关数据的处理流程也需要经常调整。此次IBM联合SAP的项目最大亮点就是“无边界”（Borderless），用户可以运行分散在不同虚拟计算机上的应用，而从宿主看这些应用可能又同时运行在一台物理计算机上的多个逻辑分区里。用户的数据访问、业务处理可以在不中断的情况下由云平台调度，并根据业务热点的需要自动机动到新的区域。

从去年发布Windows Azure之后，微软在云计算方面的表现始终迅速而优异。本月微软宣布将在SDS（SQL Data Service）原有实体对象模型（entity-based data model）的基础上，提供关系模型的访问接口。如果客户希望以REST-方式访问关系模型接口，则需要自己订制ADO.NET Data Service访问。关系模型将基于TDS协议（Tabular Data Stream），协议的细节内容会在Las Vegas的MIX 09上介绍。

MySQL被SUN收购后也开始出现不同路线问题，正如Computer World上一篇《Which SQL is MySQL?》的博客所言，尽管SUN收购MySQL后同时对外提供免费、收费两个版本，但出于商业成本和投资收益的考虑，MySQL/SUN的下一个版本似乎要经常带着很多已知或未知的致命Bug发布，而MySQL的另一个分支MySQL/MariaDB则继续秉承着之前的理念：社区贡献开发、稳定、始终免费，但MySQL/MariaDB前景如何还要时间证明。■

虚拟化

■ 主持人：刘超

华中科技大学计算机系统结构硕士，虚拟化973项目多计算系统资源虚拟化小组成员，网络工程师，软件设计师。

虚拟化的崛起

虚拟化巨头VMware终于有了自己的中文名字——威睿，“威”象征力量和威望，“睿”代表睿智和远见，表达了VMware致力于利用虚拟化技术平台和解决方案为客户提供重大价值的愿望和承诺。VMware自2005年进入中国市场以来，就取得了令人瞩目的成就。短短4年时间，VMware在中国的合作伙伴数量增长已经超过10倍。近日，VMware又与曙光达成全面OEM协议，曙光公司将销售及支持基于曙光服务器系统的VMware Infrastructure 3。

中国虚拟化技术网络大会3月份的议题是“桌面与应用虚拟化”，这表明虚拟化技术在大型服务器、数据中心、云计算等领域初露锋芒之后，正迅速进入到桌面领域。在不久的将来，我们的桌面将不再是单一的硬件设备，而是各种复杂设备和异构环境的集合。利用桌面虚拟化技术，我们就能使程序和数据，系统环境和硬件分离，并提供给用户一个统一的视图，使用户可以随心所欲地访问所需要的程序和数据，从而为用户提供虚拟化的无缝体验。

据Gartner在3月发表的一篇研究报告称，2013年全球托管的虚拟桌面(HVD)市场的销售收入将达到657亿美元，VMware、思杰、微软和其他虚拟化公司将从中受益，虚拟桌面市场的增长也将推动惠普、戴尔和其他硬件厂商采取自己的行动进入虚拟桌面市场。这预示着桌面虚拟化市场的竞争即将拉开序幕，Sun公司正在试图用xVM VirtualBox 2.1对抗VMware的桌面虚拟化产品，思杰近日对外宣布，将与英特尔合作开发针对采用英特尔vPro技术的酷睿2桌面级处理器以及迅驰2笔记本平台处理器设计的桌面虚拟化解决方案，这将是业内首款以笔记本电脑为主要对象的桌面虚拟化产品。随后，思杰还发布了新一代的桌面虚拟化产品Citrix XenDesktop 3。而VMware在推出VMware Fusion，VMware Player和VMware View等产品之后，在2008年末又发布了新版桌面虚拟化解决方案VMware View 3，今年2月更是发布了开源客户端VMware View Open Client来推动和开源伙伴的合作。各大主要的虚拟化厂商几乎同时发力，着手进行战略布局，预示桌面虚拟化已经纳入厂商的整体战略。■

游戏开发

■ 主持人：赵青

九艺科技CEO，曾任金山西山居技术总监，网易方舟产品总监，《剑侠情缘2》项目负责人及主程序员，资深游戏开发者。

九城不高兴

最近有本书很火，《中国不高兴》。我仔细的看过了，简单的说虽然有些偏激，但大部分观点我都认同。比如“内修人权，外争族权”，两手都要抓，一个都不能软。另外，强势地区对弱势地区的轻视是一个客观事实，加上国内的一些人盲目的崇洋媚外，使得一些外国人真的被惯坏了。

如果这几天的消息属实：《魔兽世界》的运营权要转交给网易，那九城真的要不高兴了。在没有最终明确的消息时，本不该写下这篇文章，但我实在是忍不住，这件事情太有意思了。

《魔兽世界》大陆地区的运营权易主已经传了很久，一方面种种的传言，以及两家公司的动作都说明这件事很有可能。比如，《魔兽世界》最新资料片《巫妖王之怒》的审批一直不能通过、网易和暴雪在国内成立合资公司，以及据说是九城CEO陈晓薇的一份内部邮件等等。但另一方面，大部分人都不相信这件事情，原因是《魔兽世界》对九城太过重要，失去这个产品后九城会元气大伤，这个事实我们知道，九城也一定知道，所以其他竞争《魔兽世界》代理权的公司能接受的条件，九城也一定也能接受，加上九城在《魔兽世界》的运营上已经积累了相当的经验，相同的条件下，暴雪也一定会选择九城。

个人觉得如果这个传言最终被证实，那么唯一的赢家是暴雪，九城和网易则是双输的结果。九城受损毫无疑问，网易虽然得到了一个明星级的产品，但付出的代价一定超乎我们的想象。看到国内运营商这么争夺《魔兽世界》的代理权，暴雪一定在偷乐。这和几年前国内运营商疯狂抢夺一些所谓的韩国网游大作的情景多么类似。其实，国内网游的研发水平这几年提高很快，如果只针对大陆地区的玩家，自主研发有很多的优势，大可不必对国外产品盲目崇拜。需要说明的是，暴雪是我最欣赏的一家游戏公司，《魔兽世界》也是我唯一觉得有意思的网络游戏。即使这样，从市场的角度看，《魔兽世界》也不是大陆地区最好的网游产品，《梦幻》、《天龙》这些本土研发的产品，在市场占有率和营收方面都超过了《魔兽》。而且我相信本土、自主研发的网游会越来越好。■

安全

■ 主持人：肖新光
安天实验室首席技术架构师，研究方向为反病毒和计算机犯罪取证等。

互联网巨头引发的隐私恐惧

4月1日，关于“谷歌”建立了一支由携带探测器的“谷鸽”组成的探测网络的消息在Google首页上公布，消息中说大量带有信息采集器的鸽子，组成了一个谷歌山寨信息网。当然，这是一个愚人节玩笑。但其反应非常连锁，很多人立即感觉到一种毛骨悚然的寒意。事实上，最近两个月，Google的街景服务就已经引发了关于个人隐私的争议，也因此删除了数百张图片，因为大家在街景服务中清晰的看到了诸如人们从成人用品商店中出来的情景，而这种取之于民，用之于民式的公共服务的巨大安全威胁，就在于其广泛的信源所带来的超越传统国家和政府安全机制的体系，从而将美国的战略威慑隐然于少数巨头的企业价值当中。

本月，另外引起争议的是输入法的混战，从逆向窃取词库到相互屏蔽的口水官司，我们已经看到了很多，本月的争议点则在于出品商是否再窃取用户输入词库。若干年前，微软拼音的作者哈工大一位教授曾嘲笑某输入法的思路是百万词大辞典。但今天来看，在人肉的巨大威力里，整句识别的AI黯然失色。但从安全角度看，一个更大的威胁是，输入法用户词库的上报不仅带来一种知识产权争议和隐私威胁，更是一种潜在的窃密和定性目标节点性质的通道。只要对用户输入的词频进行统计，就有可能定位用户职业习惯等，从而引发关联泄密事件，此外，部分输入法捕捉用户剪贴板的特性也有可能被利用。这些都值得警惕。

本月，最大BT站点海盗湾创始者被判处有罪，这是知识产权保护史上的标志性事件，我们已经看到了水印技术的少数倡导者又在以此说事，再次把视频水印的话题提出，这是有点“啼笑皆非”。其真实价值暂且不谈，在鲁棒性等痼疾解决之前，这项技术恐怕更多的还是paper的贡献。■

商务管理解决方案

■ 主持人：贾菡
解决方案专家，曾获微软Dynamics CRM产品认证专家、微软CRM MVP称号。有多年企业信息规划管理、企业商务管理解决方案的项目经验。

医改，颓势中的强劲机会

4月6日，中国新医改方案正式对外颁布，在这部被比喻为“一顶、四梁、八柱”的医改文件中，史无前例地用一个专门章节对信息化提出讨论，并将之列为“八柱”之一；5项改革目标全部都涉及到利用信息化为政策推行实施提供支持，这是外国软件公司直接得益于中国经济刺激方案的少有机会，更重要的，它必将会给现在相对低迷的软件业带来大量机会。

4月8日，IBM大中华区软件集团与其中国开发中心（CDL）共同宣布成立“IBM医疗行业解决方案实验室”，公布了IBM智慧医疗系列解决方案。IBM中国开发中心认为中国在医改投入的8500亿元人民币（1240亿美元）中将至少会有15亿美元的软件支出。因此IBM将当仁不让地利用这次难得的机会抢占医疗行业的市场，现在已经开始执行第一份此类合同——为中国最大的中医院广东省中医院提供一套电子病历系统。

这不禁让我们回想起比尔•盖茨创建基金会并捐出个人全部财富的主要目的，是为了解决“全球健康问题”。微软中国战略之父克瑞格•蒙迪也表示教育医疗业务将成微软业务发展的新重点。2006年微软成立了医疗事业部，并通过一系列收购和资金投入来壮大其在这一行业的力量。其后推出的“Amalga”这一针对医疗行业的系列信息技术平台和解决方案，用以构建全新的数字医疗，以连接起不同医院及同一医院不同部门间的“信息孤岛”。同时，在“软件+服务”理念的指引下，微软还有服务于普通消费者的“Microsoft HealthVault”网站和医药信息搜索引擎“Microsoft Medical Search”，它们和Amalga通过互联网连接后，将为医疗系统以及普通用户提供数字化医疗服务。另外微软还通过其CRM系统为用户提供医疗行业的解决方案。

Google紧随微软之后，推出与Microsoft HealthVault相抗衡的线上健康医疗业务Google Health，这也是互联网大鳄和软件巨人在数字医疗服务领域正式交锋。

此外还有在生命科学方面领先的Oracle，专注于医疗行业流程支持的SAP以及了解国内医疗体系特点的本土软件公司，相信都会在推出这项新方案时为之一振，在各自擅长的领域抢夺先机。■

动态语言

■ 主持人：高昂

资源与环境信息系统国家重点实验室博士生，关注动态语言，喜爱写旅行游记，同时也是OSGeo中国和InfoQ中文站成员。Blog：http://www.gaoang.com/

JavaFX的挑战与机遇

JSR 223规范为JVM加入了动态语言运行时的支持。当动态语言代码转换为字节码运行在JVM之上，可以方便地享用Java平台在拓展性、移植性和安全性等方面的诸多优势，同时可以引入了Java API和为数众多的第三方库。JavaFX是运行在JVM上的脚本语言，自两年前在JavaOne大会诞生算起，已经由最初Chris Oliver创建的实验性项目F3演变成一套完善的RIA解决方案。

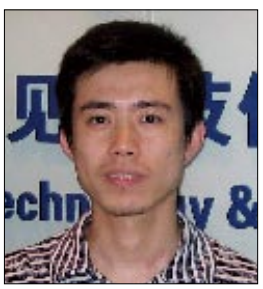
在语法上，JavaFX使用专门设计的DSL对GUI界面进行描述，融合了JavaScript、ActionScript等脚本语言的特征，并且同Java代码有一定相似性，但较之Java代码抽象层次更高，且直观易读。同时，JavaFX文件简短，无需其他RIA实现冗长的XML界面描述，对习惯于编写代码的开发者来说，完全可以脱离设计器手工编写JavaFX应用的界面。

较之Flex、Ajax、Silverlight等主流RIA技术的脚本语言，JavaFX Script起步虽然较晚，但在技术上却有独到之处。除JavaFX SDK提供的通用特效、动画功能之外，代码中可以直接调用Java API和第三方Java类库。应对桌面开发，可借助Java多线程特性来保证应用的健壮性。尽管JavaFX并非为取代Swing界面而出现，但有开发者已开始尝试在Java代码中反向调用JavaFX定义的类，以便在应用的MVC分层中，表现层使用JavaFX，逻辑和控制层由Java代码实现。

在JavaFX目前最新的1.1.1版本中，JavaFX Mobile提供了移动终端支持，借助于JavaME的广泛应用，JavaFX在移动终端平台的优势要强于Flex和Silverlight等RIA技术。

但值得我们注意的是，JavaFX代码通过自身Class Loader加载到JVM上运行的效率还需要进一步优化，一般不大的JavaFX应用从装载到启动的时间仍然相对较长。在IDE支持方面，NetBeans提供的JavaFX模块已集成了不少可拖拽添加的JavaFX组件，但在用户界面控件上较之Flex和Silverlight的开发工具仍然相对薄弱。不过Sun已在计划JavaFX的2.0版本中提供完整的用户界面组件，包含容器和布局控制等不同层面。■

嵌入式开发&硬件



■ 主持人：孙天泽

华清远见嵌入式培训中心金牌讲师，嵌入式行业资深专家，畅销书作者。

英特尔宣布嵌入式处理器

2009年春季IDF峰会在北京如期召开。不知不觉中，英特尔IDF已在中国举办了十年，在这十年中，英特尔为我们奉献了无数场创新技术与产品盛宴。即便是在金融风暴大环境下，英特尔依然在为技术创新不断努力。

本届峰会上，上网本是当仁不让的热点。去年的IDF峰会上，英特尔推出了凌动处理器，也许当时英特尔自己都想不到凌动能如此成功。在今年的IDF主题演讲当中，英特尔移动互联网事业部总经理阿南德又推出了两款最新的凌动系列处理器，给了上网本产品更强劲的性能。新凌动具有更高的主频，更低的功耗，所有指标都显示了英特尔要垄断上网本市场的决心。与此同时，英特尔还宣布了另一款主要针对存储设备以及嵌入式产品的处理器——Jasper Forest。Jasper Forest将有多种版本，从单核、双核到四核心，以适应不同嵌入式产品的应用需求。

从市场表现看，市场嗅觉灵敏的英特尔再次抢占了先机。但是作为半导体领域的老大，英特尔只是做了该做的事情。倒是由于上网本的走红，使得不少PC厂商从中获益。在本月举行的2009中国国际数码通信展（DigicomeChina2009）上，上网本专区成为了最热闹的地方。长城、神舟、华硕、中柏、华旗资讯等纷纷推出了最新的上网本产品。

除了英特尔，另一个对上网本处理器虎视眈眈的是ARM公司。占据嵌入式芯片市场半壁江山的ARM公司始终没有杀入PC市场的确是种遗憾。不过随着上网本3K Computer公司的3K RazorBook 400上网本的面市，ARM公司进军PC的战役已经打响。3K Computer公司的这款本只需155美元，是最便宜的上网本，使用了ARM处理器，安装了微软的Windows CE操作系统。虽然配置不高，但是却严格符合上网本的定位，与现阶段的12、14寸高性能笔记本划分了清晰的界限。有消息称，飞思卡尔和高通都准备在今年6月的台北Computex 2009大会上发布各自的ARM平台上网本产品。其中飞思卡尔的上网本采用基于ARM Coretex A8架构的i.MX51处理器。ARM能否凭借上网本进军PC领域，答案将在今年暑期揭晓。■

企业级开发

■ 主持人：胡长城

胡长城，网名“银狐999”，普元软件R&D部门架构师。在Workflow、BPM、SOA领域有过多年的产品研发和实施经验。

企业应用开发

出差外地，错过了4月初的两个比较重要的开发者会议，QCon和Adobe开发者技术日。笔者最想了解一下以Flex为代表的RIA应用体系以及对企业应用开发所带来的影响。但时至今日，网上有关这两个会议的一些PPT尚无多少可show的，这充分的展示了主办方“售后服务价值及意识”的缺失，事实上，企业应用开发领域，售后服务也很重要。

在最近的SOA中国论坛，普元软件发布了国内第一款支持最新SCA/SDO标准的SOA应用开发平台，提出了“SOA从应用开始”这个符合国情的“SOA之路”理念，正视了国内外SOA应用特色的差异化。国外整体的企业应用环境比较成熟，应用产品的模块化程度普遍也比较高。所以围绕SOA应用的架构和产品体系主要是以“集成”及“总线式解耦”的模式来运作。而国内的企业应用环境尚处于起步、发展阶段，早先的模块化成熟度也比较低，所以客户围绕SOA应用需求主要还是以“应用构建、服务暴露、服务装配”为主，这样就形成了国内外SOA应用需求和模式的差异化。

3月末，RedHat发布了JBoss Developer Studio 2.0——Portfolio Edition，它不仅增强了Eclipse工具集，还添加了JBoss Enterprise Application、Portal、SOA与Data Services平台。该解决方案Rick Web应用和企业应用与集成服务提供了健壮、集成的开发环境。开发者可以凭借其使用Seam、Java EE、AJAX及Portlets等面向服务的架构快速开发交互式的应用和服务。

TechTarget所发起的一项关于BPM和SaaS趋势的调查表明，尽管年初出现了一些关于SOA的负面报道，但事实上SOA依然在大踏步前进着。有49%的受访者说其公司正在进行着一个或多个SOA项目，60%的人表示其当前或未来的SOA项目将成为企业级而非部门或分支级别的项目（21%）或是独立的项目（19%），同时这些受访者也承认还有很多困难阻碍着SOA的发展。

3月底，金蝶发起了“中小企业全程电子商务服务联盟”，通过提供全程式的在线管理服务和在线协作服务，来解决中小企业电子商务问题，同时为SaaS应用模式寻找出新突破。■

RIA技术

■ 主持人：吴建伟（花名独行）

淘宝PMUE - UED - 前端架构工程师，网名傻鱼，“傻”是我的生活哲理，执著是我的信念。

SQL离前端有多远？

SQL全名是结构化查询语言（Structured Query Language），一直是后台开发者用来操作数据库的语言。对于我们大部分前端开发者来说，数据库和SQL目前看似都是遥远和陌生的。

然而随着Web的发展潮流，一方面Web正在成为新的应用平台，越来越多的Web App为了提高用户体验，纷纷推出离线功能，比如Gmail、Facebook、WordPress等，当然它们都依赖于本地存储方案。W3C在最新的HTML5草案中，就加入了浏览器端本地数据存储（Web Storage）的规范，而SQL语言就是其中的一部分，这对RIA应用有特别重大的意义。浏览器内置数据库，通过脚本操作客户端的本地存储，就可以圆满的解决如何保持客户端状态的大难题。让人欣喜的是，尽管浏览器大战狼烟四起，但是新版的主流浏览器都不约而同的提供了对本地存储规范的支持。

另一方面，云计算的出现，让OpenAPI的应用越来越广泛。看看Yahoo！的YQL，以及Facebook的FQL，还有Google App Engine的GQL，三个业界翘楚推出的基础开放平台上的结构化数据的查询语言，都用了大家熟知的SQL-Like的语言作为自身开放平台的接口交互语言。

我想原因有以下几点：SQL非常易学，它是大部分Web开发者都已经熟知的一个语言标准，很容易上手；SQL语法简单，但应付一般的数据查询和操作已经绰绰有余。特别是对于轻量级的本地存储来说，直接用SQL已经能满足绝大部分业务需求，同时非常便捷；用SQL的语法，可以让OpenAPI的设计优雅简单。

我们几乎可以肯定，随着后台程序的底层服务化，除了负责数据展现，数据的获取和操作也会逐渐成为前端工程师的工作内容。这也意味着前端工程师的职责会越来越大。当然，我们需要掌握和了解的技能自然也会更加丰富—HTML、CSS、Javascript、Flash、服务端编程语言 … 而SQL正在加入到这个序列中来。

SQL语言离我们前端不远——它，已经轰然在我们眼前！■

摩尔定律将持续发挥作用

——IDF 2009大会纪实

英特尔公司董事会主席克瑞格·贝瑞特亲自从美国赶赴现场

■ 记者 / 欧阳

十年前，英特尔在中国举办了第一次信息技术峰会。尽管参加过那次论坛的人并不多，但通过这次活动，让中国用户第一次有机会直接接触到个人电脑时代最伟大的公司之一——英特尔。十年只在倏忽之间，而英特尔信息技术峰会给人带来的好奇与新鲜，却始终没有停止过。

聚信与共，创赢未来

4月8日，英特尔信息技术峰会（IDF）在北京召开，其主题为：“聚信与共，创赢未来”。来自各个行业的IT决策者、开发人员以及英特尔资深管理人员和技术专家共同参加了此次大会。

英特尔公司董事会主席克瑞格·贝瑞特亲自从美国赶赴现场，为大会带来了第一场总揽全局的主题演讲：“鼓舞心智的创新”。他在演讲中提道：“过去这些年，IT领域充满了激动人心的时刻。晶体管诞生于60年前，集成电路已经50岁了，PC已经走过了37年的历史，因特网经历了20个春秋，而类似Facebook这样的社会网络，还仅仅只有三四岁，整个IT领域在持续创新，而创新的根本则是技术上的不断进步。”

回顾十几年前的英特尔，从奔腾时代开始的技术创新就一直持续不断，包括在奔腾Pro上为英特尔打下江山的MMX技术、应用于奔腾III的SSE技术等；多核时代到来后，英特尔更是采用其酷睿2架构，帮助开发者设计并行程序，同时推出Ct平台等并行计算的开发环境。贝瑞特表示，摩尔定律，在未来的15年内仍然会继续发光。在本届IDF上，最耀眼的则属其最新公布基于32纳米的Nehalem架构，这将进一步提高其处理器的并行计算能力。

随后，英特尔高级副总裁帕特·基辛格展示了未来几年英特尔处理器技术上的发展路线图：从2009年开始，我们将进入真正的多核时代，四核处理器在今年将成为业界的主流产品，32纳米的处理器工艺也将逐渐走向用户。包括在上月正式发布的至强5500企业计算平台将会给互联网和企业级计算的基础设施提供更强大的运算能力。随后，我们将很快就可以看到8个内核的处理器，英特尔也会在2011年年底之前推出22纳米的处理器工艺，届时，新的处理器将会采用更先进的Sandy Bridge架构，从而满足更大的计算需求。

英特尔的移动战略

除了通用平台外，主题演讲还包括了英特尔副总裁阿南德对移动方面的阐述。面对快速发展的低成本上网本市场，英特尔率先推出了其基于便携计算设计的Atom处理器。除此之外，在软件平台领域上，英特尔还展示了其最新发布的开源产品Moblin 2.0项目。这些产品的展示与发布，表明英特尔在中国即将迈向3G时代的同时制定的清晰战略。

英特尔对中国市场寄予厚望

从这次大会上，我们可以看出英特尔对中国市场寄予的期望。在贝瑞特的演讲当中，除了为我们展现技术创新给时代带来的变化，也同时关注到了一些目前中国市场的重要方向。一方面，英特尔大量强调了中国的农村市场。可以毫不含糊的说，目前中国的农业人口居全球之冠，而如何充分有效地挖掘这个市场的潜力，对英特尔来说具有很大挑战。另一方面，贝瑞特还强调在医疗行业的应用上，中国还有很大的发展空间。庞大的人口基数在面对全面规划的同时，也使IT企业大笔获得政府投资的最佳机遇。

同往年大会做派不同的是，今年的IDF被压缩为一天举行。由于受到整个经济环境恶化的影响，英特尔也在此时收紧了其在市场活动方面的预算。紧凑的议程让整个大会的气氛非常活跃，尤其是在中国亟待开发的市场，用户的反馈显得更加强烈。■

英雄的会，你的会

——记2009中国软件技术英雄会

■ 记者 / 圣伟

CSDN &《程序员》总裁蒋涛致开幕词

4月18日，由CSDN&《程序员》主办的，2009中国软件技术英雄会（北京站）盛大召开。此次英雄会百名CTO、架构师、高级软件开发人员、技术作译者以及CSDN社区技术英雄汇聚一堂。主办方力求在一天的会期中，聚焦于“创业、创新、创富”话题，通过主题演讲、电梯游说和论坛互动等环节，向参会者呈现前沿技术、主流趋势、高端视点及技术创业要诀。在这个乍暖还寒的冬春之际，燃起一把IT技术圈之火。

大会伊始，CSDN&《程序员》掌门人蒋涛致开幕辞：“现在，我们正经历一场变革，一场技术与平台的变革。这其中包括移动终端的崛起，互联网平台的巨变等。这场变革，可能是有史以来最重要的。所以，今天我们将一起分享和讨论它对我们每一个人、每一家公司的发展所带来的影响。”

“嘤其鸣矣、求其友声”。在英雄会上，与会者将相识各路精英，开拓新机会、了解新趋势、分享技术经验和展示创新成果。正所谓“英雄的会，你的会！”

主题演讲，你方唱罢我登场

在随后进行的CSDN&《程序员》颁奖典礼上，主办方颁发了CSDN&《程序员》最具特色的几项大奖：2008-2009年度中国软件金牛奖、2008-2009年度CSDN最有价值专家（CSDN MVP）、2008-2009年度CSDN最有价值博客（CSDN MVB）、2008-2009年度CSDN最有价值版主。这些奖项，不仅是对在“寒冬”中顽强拼搏的各大互联网、软件公司的一种鼓励，鼓励他们创造出更多更好的软件产品，也是对在CSDN社区和博客中，勤于分享和交流的“思想实践者”们的赞赏和肯定。

颁奖典礼结束后，主题演讲的环节便开始了。虽然演讲与论坛交错进行，但安排十分紧凑，俨然呈“群英荟萃，你方唱罢我登场”之势。

微软大中华区开发平台合作部总经理Nigel Burton作了主题为“软件•社会•创新”的开篇演讲：“全球有60亿人口，但受益于IT技术的人却不到10亿，当我们享受身边先进的，给我们带来方便的IT技术时，更应该思考如何利用它们去帮助整个社会。目前，并没有很多人利用移动终端设备上网，而在较为偏远的城乡，甚至很少有人能够接触互联网；中国有上百万的学校，而信息化教育却不是那么普及。所以，我们要用越来越多的新技术去解决这些问题。”的确，利用软件技术实现全民信息化的社会责任感，是我们进行技术创新的原动力之一，也是进行创业的一个良好切入点。

在此之后，谷歌开发技术推广部中国首席经理栾跃由Google的云计算开发平台的发展战略谈起，生动而深

入地分析了云计算开发平台和其开发的商机所在；康盛创想（北京）科技有限公司漫游运营部经理程韬以什么是漫游开放平台为序幕，描绘了Social Game（社交游戏）的未来；北京糖果网络技术有限公司负责人唐爱平，也给大家带来了主题为“新一代浏览器的战争”的精彩演讲。

最后，Digital River公司资深新兴市场总监施文祥先生，为大家带来了主题演讲——《软件技术人员创富之路探析》。他从分析成功创业者的性格特征入手，解读了为何性格才是软件技术人员在创富之路上成功的第一要素：

“如果你希望有一个正常的工作、生活时间，那么就不要从事编程的工作了。此外，你是否具备足够的抗压和抗挫折能力，也决定着你在这个行业内的发展情况。作为软件开发商而言，管理者也需要不畏惧困难的激情和热情。”

三大论坛，英雄过招

本届英雄会上的三大论坛——架构师论坛、创业论坛、产品论坛，既给了诸多业内专家一个各抒己见的机会，也是与会者与专家“过招”的交流沟通平台。

在架构师论坛上，来自西门子中国研究院软件与工程中心的首席系统架构咨询顾问李伟、北京南天软件有限公司研发总监及首席软件架构师王玮、新网互联科技有限公司研发总监王泽宾、RedHat（红帽）中国首席架构师薛伟，均到场为与会者分享了诸多关于架构师的观点的从业经验，使与会者感慨颇深，解除了长久以来心中的疑惑。

在讨论中，薛伟将架构师分为两类：第一类是系统架构师，他所关注的核心领域是应用；第二类是应用架构师，是规划业务的，包括业务的逻辑，应用架构里用到的所有底层技术等。他认为，架构是一种开发的意识，开发的架构实际上被认为是Frame Design，就是开发框架式的结构设计。

随后而来的创业论坛中也是创业英雄云集，康盛创想（北京）科技有限公司CEO戴志康作为80后创业的成功代表，用其诙谐幽默的风格讲述了他关于创业的一些看法和他自身的创业故事。迅雷联合创始人兼CTO李金波的一句话代表了他关于创业的观点，并且让人难忘：技术本身很重要，但如果不能商业化，就没有任何价值。

2009中国软件技术英雄会（北京站）架构师论坛现场

在最后进行的产品论坛上，诸位英雄各抒己见，共同讲述了一个好的产品是怎样炼成的。当谈到好的产品是怎样定义的这个问题时，Google（谷歌）产品经理杨巍表示，好的产品要有好的理念来支撑。Google的很多产品都采用了云计算的理念，这一点满足了用户在使用不同终端时，可以始终保持自己特定习惯的需求，使产品的生命力变得更为持久。

而IJI中国公司董事总经理吴穹认为，独占性是可以证明一个产品是否足够优秀的显著特征，如果一个产品占据了市场的绝大部分份额，那么一定是个好产品。

IBM中国软件开发中心Rational总经理严成文和现任奇虎360公司产品经理的马占凯也各自发表了自己的观点，前者认为简用和易用是好产品需要具备的特点；后者则认为，好的产品，其基础性能指标一定要好。

电梯游说，挑战三分钟的激情

本届英雄会延续了去年非常成功的5分钟电梯游说，但是将时间缩短到了3分钟。这就说明，到场的几位创业者要在更为短暂的时间内，把自己的项目和产品，用最有效的方法介绍给风险投资商。

虽然，众多创业者还不能完全适应这种极富挑战力的演讲模式，但他们的演讲技巧还是在该项目中，得到了充分地锻炼和最大程度地发挥。蒋涛在会后的博客中发表了自己的观点：电梯演讲和创新技术展示的绝大多数人演讲技巧亟待提高，如何在相对短的时间里把自己产品和技术的优势展示出来，吸引用户和投资者，这在以后会越来越重要。糖果的唐爱平是典型，很容易陷入到讲解技术细节和全面分析之中，而短时间的演讲，听众其实不可能理解这么多，重要是讲清楚你能做什么？你的产品和技术如何帮助用户解决问题？然后演示出来给大家看，有清楚的数字或者生动的展示就非常好。

写在最后

对更多不能在演讲台上展示的英雄们来说，参与英雄会最大的收获之一还在于结交同道，交流彼此的技术和各种心得。软件技术英雄大会，是CSDN&《程序员》杂志提供的一个平台，在这个平台上每个人都是主角，每个人都是英雄！

三日飨宴，回味无穷

——2009 QCon北京站侧记

InfoQ总编Floyd Marinescu、中文站总编霍泰稳

从左至右：Dylan Schiemann、Henrik Kniberg、Rod Johnson、Martin Fowler、王文彬、Randy Shoup

■ 记者 / 郑柯

InfoQ是近几年异军突起的软件开发技术网站，除了在网站上提供高质量的技术内容外，InfoQ还因其组织的技术大会——QCon而在业界知名。2009年4月7日至9日，QCon全球企业开发大会在北京清华科技园国际会议中心举行。此前，该会议仅在美国旧金山和英国伦敦出现过，这次也是第一次在亚洲举办。

讲师：群英荟萃

作为一个由媒体主办的技术大会，2800元的门票价格有些令人望而却步。然而出乎很多人意料的是，本次大会门票竟然全部售罄。即便如此，到会议现场购票的人仍然络绎不绝。究其原因，前沿的技术话题和高水平的讲师是最大的吸引力。在旧金山和伦敦举办的QCon，Erich Gamma与Kent Beck都曾是其座上客。本次QCon同样有如雷贯耳的大师级人物，比如ThoughtWorks首席科学家兼Agile领域伟人Martin Fowler、Spring框架创始人Rod Johnson、eBay核心架构师Randy Shoup、Dojo Toolkit联合创始人Dylan Schiemann等。

国外讲师固然声名显赫，国内的讲师同样实力非凡。他们有的来自大公司，比如淘宝首席架构师王文彬博士、IBM中国Web 2.0首席架构师毛新生，有的则来自新兴Web2.0网站，比如豆瓣网首席架构师洪强宁、优酷网核心架构师邱丹，也有像台湾软件架构设计大师高焕堂、技术作家兼资深架构师周爱民这样的观察者兼行动者。他们同样为本次大会带来了令人激赏的议题。

议题：目不暇接

本次大会分6个主题：“企业级Java开发”、“敏捷在路上”、“云计算——下一代架构”、“网站架构案例分析”、“设计优良的架构”和“RIA——炫富互联网应用”，共33个session。敏捷和架构方面的内容，一直是QCon的亮点和长项。本次在北京站也不例外。

《硝烟中的Scrum和XP》是备受国内敏捷开发爱好者喜爱的一本小书，很多人都以其作为实践敏捷的参考书。此次QCon上，本书的原作者Henrik Kniberg带来了“多团队的Sprint计划”的议题，同样以其深入而独到的实践性内容让观众收获良多。

而豆瓣网首席架构师洪强宁关于网站架构发展历程的演讲，被众多观众交口称赞，有观众会后在博客中这样评论：“该Session按照时间顺序组织。从豆瓣一开始上线的情况、架构选择，到后来不同用户级别时候出现的不同问题，应对措施，解决办法，甚至相关代码。非常完整和清晰的展现了豆瓣架构发展里程和其中所采用的具体策略。这样具体的Topic非常精彩。强宁的演讲技巧也蛮好，语速发音都很专业，一定是练过。有趣的话题，专业的演讲，加上得当的Slide，个人认为本Session可以当之无愧成为QCon今年的最佳Session。”

除此之外，大会第一天晚上的大师论坛同样备受关注。该论坛由InfoQ总编Floyd Marinescu主持，话题围绕“企业软件开发的趋势”而展开。在Floyd一个又一个问题的轰炸之下，各位大师脸不变色心不跳，你有来言我有去语，场面异常火爆。不过，诸位大师话语之中有一个反复出现的关键词——开源。大家都认为：在目前的经济形式之下，开源已经成为企业降低IT成本的首选；同时，再搭配敏捷开发方法和实践，企业就能够迅速提高投资回报率，并同时让客户的满意度得到提升。

未来：更加精彩

虽然本次QCon得到诸多好评，然而还是有些许不足之处，比如某些国外讲师的话题无甚新意，而某些组织和后勤方面还可以做得更好。但考虑到InfoQ中文站只有3个全职人员，本次QCon能够做到如此程度，已经令人颇为赞叹了。

看到如此热烈的气氛，Floyd表示：明年的QCon一定会更令人期待、更加精彩！

开发者是至关重要的角色

——微软商业软件业务部总裁Stephen Elop专访

■ 记者 / 欧阳

早在2007年年底的时候，微软就以S+S这个名词给整个软件领域定义了全新的解释。由于互联网时代的到来，整个软件商业环境变得越来越混乱。新模式下，更多的人在讨论产品、应用、服务、项目、商业模型等业务话题，而原本在整个软件领域处于核心位置的技术与开发，则似乎正在被逐渐淡漠。而关于这件事，微软商业软件业务部总裁Stephen Elop却有非常不同的看法。

开发者是至关重要的角色

在采访中，Stephen认为："在我们继续推进'软件+服务'战略的过程中，开发者将会扮演一个至关重要的角色。其实在技术的衍生和发展的过程当中，我们可以看到，对于每一代新的技术而言，只有开发者的团体能够真正的接纳它，容纳它，这个技术才能真正的获得成功。这也是为什么我们微软在'软件+服务'一个最大的消息就是在去年晚些时候在微软面向开发者的一个专业会议(PDC)上宣布的。这也充分证明了我们对于开发者群体的重视。"

尤其是在云计算的时代，Stephen表示开发者的作用，尤其是微软技术的开发者，将会发现自己的地位更加重要。他补充道："比如说我们有一个叫做Windows Azure的底层的平台，开发者可以利用这个平台做他们自己的开发，如果能够走这样的道路，他们就能够充分使用微软在过去一直以来开发出来的和一直使用的技术和技能，并且保证这些技术和技能在未来新的世界中依然有效，并发挥作用。当这些开发者在使用这些新平台及新服务时，就意味着他们也在同时使用微软过去一直以来就已经有的，并且为他们使用的工具、技术的进步，都会是有效的。"

Gartner副总裁内尔•迈克唐纳德(Neil MacDonald)表示，微软又一次掌握了主动权，而这也是整个行业发展的趋势。虽然亚马逊和Salesforce.com在云计算市场上走在了微软的前面，但微软仍具有优势。微软同第三方开发者有着长期、深厚的关系，并且在搭建软件平台上要更具有经验。

然而在S+S战略的制定和实施过程中，整个进程却显得稍微有一些滞后。Stephen解释道："战略的执行在中国还处于非常的早期，包括在全世界也是这样，现在有一些服务在美国已经有了，大概两星期以前，我们在世界18个国家推出了这些服务(BPOS)，这个工作在以后肯定要继续的，而且还有大量的要做的工作。微软的团队正在试图理解中国市场，以便以比较好的方式在中国推出在线服务。这些工作包括，比如要了解政府的法律法规和监管的政策，要理解在中国做这些工作本土的成本情况等。"

投资的三点建议

为了更好的投资于开发者市场，Stephen总结了三点：

微软商业软件业务部总裁Stephen Elop

第一点，实际上微软已经做了大量的投入来专门针对开发者提供一些项目，尤其是培训的项目，帮助他们更好的理解和掌握微软技术，这包括在线的MSDN以及线下的活动和社区等。

第二点，微软确保能够为所有的开发者们在尽早的阶段提供接触到最新技术的机会，包括服务器产品、开发工具以及各种编程模型在内的社区版本都会先于市场推出。

第三点，在进一步提升云计算的过程中，会在互操作性上做更大的努力。一年前，微软已经承诺就最成功的商业产品实施互可操作性的原则，这些产品包括Exchange、Sharepoint、Windows、Office等，为开发者提供了非常多的机会，确保他们自身开发出来的应用和解决方案能够和微软的软件环境完美的匹配，这对开发者来说是一个巨大的支持。■

思考者与行动者

——专访西门子中国研究院首席系统架构咨询顾问李伟

■ 记者 / 郑柯

在刚刚结束的英雄会和QCon上，作为架构师的代表，李伟发表了有关系统架构的演讲，并参与了相关讨论。如果认真听过他的发言，你会发现：与其他人相比，他的很多观点有些“格格不入”。当然，这与他的个人经历密不可分。

李伟，西门子中国研究院软件与工程中心首席系统架构咨询顾问，西门子核心专家及业务主要负责人。他具有丰富的系统架构/设计、开发和咨询的相关经验，曾参加过企业信息化系统，工业实时系统、嵌入式系统、分布式系统整合和消息系统等多种项目。他为西门子内部各个商业业务集团和其他外界商业公司的客户提供系统架构的全方位咨询和服务工作，包括西门子的工业自动化集团、楼宇集团、交通运输集团、西门子软件开发中心、Motorola、诺基亚·西门子、中国移动总公司、中国铁道部、中国民用航空、中国国税总局、中石化总公司和国家统计局等等。这次采访，就从西门子研究院开始。

对软件的投入，西门子名列前茅

《程序员》：您能否简单介绍一下西门子研究院的情况？在软件方面有哪些贡献？

李伟：西门子研究院的发展历史比较长。当时西门子正在发展成长的时候，就意识到一个问题：核心技术方面应当要有核心的技术部门，一个中央的研究部门，这对公司战略是很好的提升和帮助。各个业务部门的研究肯定都是聚焦于各自的领域和要求，但是整个公司的成长，必然得有一些人为公司的整体利益做贡献。西门子研究院最高的院长，是西门子管理执行层的成员，级别是很高的，是西门子董事会的成员之一，同时西门子研究院的院长也是全西门子的CTO。


在大型外企的多年工作经验，让李伟有了广阔的视野和超乎常人的眼光

从人员分工和结构来看，西门子研究院划分成了很多专业技术方向。比如说这个软件与工程这个方向，实际上就是全西门子范围里头级别最高的研究和技术部门。一些国际软件方面的标准制订，是由我们这个部门的专家来参加的。比如说Sun引领的Java标准，就是由我们这个方面的工作人员来参加国际标准的制订。

软件与工程是其中的一个方向，我们也还有其他的方向，包括材料、生产制造过程、项目管理，还有系统工程、医疗方面等，基本上是根据西门子的业务来切分的。当然有一些部门明显是公共部门，比如说软件与工程，就是服务于全西门子的。具体软件的研究方向，主要有软件架构、软件开发平台和框架、软件及系统开发流程、软件开发技术、还有一个普适计算，就是大规模计算机技术在社会中的应用。比如将来到处都布满传感器，便于能够海量计算人类的活动，从而进行优化。

《程序员》：那么软件在西门子公司内部占一个什么样的地位呢？

李伟：西门子的研发人员当中，大约有超过1/10的人在做这个软件方面的工作。虽说西门子是一个产品型公司，但在软件方面的投入是很大的。现在很多产品，如果脱离了软件实际上就缺少了活力，而西门子的一个核心竞争力也体现在软件方面。

举一个例子，西门子的医疗系统。如果大家从外观来看，医疗系统好像都是一些设备、硬的东西。但是西门子的工作人员都知道：医疗系统当中很重要的方面，就是软件的复杂性，图形图像处理的复杂性。再比如说西门子的楼宇自动控制系统，从自动控制这几个词当中就能知道它是以软件为核心的。从这两个例子就能看出来西门子对软件的依赖。几年前加入西门子的时候，我就知道：对软件的投入和软件从业人员的数量，西门子在全球都是排在前几位的。

要做思考者兼行动者

《程序员》：从一个西门子中国研究院高级软件专业人员的角度来看，您认为软件架构和设计的质量应该如何评价？

李伟：业界有两种标准的做法：一种就是公司制订标准问卷进行标准评审，全方位提问，全方位搜集答案，全方位分析答案得出质量方面的评价。这是IT业界软件工程领域都知道的一个做法，叫问卷式评审、架构评审或者软件质量评审。另外一个标准做法，这个是当年美国国防部出资，卡耐基·梅隆的软件工程研究所（SEI）研究得出的一个标准做法，叫ATAM，也就是Architecture Tradeoff Analysis Method。ATAM当时首先应用到了美国航天局，到现在为止这还是一个很经典的做法，是基于场景的架构评审。对评审人员的专业水平也就是架构水平要求非常高。这些人知道解决一个问题最经典的做法是什么样，衡量你的做法好不好没有任何商量的余地，他们是权威的。所以只适用于这种场合，就是评审跨行业的、震撼性的、一些非质量功能方面的要点，可以这样做。另外还有一种比较少用的评审方式，即模拟验证。但前两种是最经典的评审方式，目前还没有其他更好的评审方式。

《程序员》：作为一个资深的系统架构师，您觉得架构师最重要的素质是什么？

李伟：作为架构师，或者说是工程界的工作人员，要尽量成为思考者和行动者的结合。有时候是沉思者，有时候就应该是强有力的行动者。要成为这两个方面的人，这才比较容易完善整个形象。如果一辈子是思考者，就有中国古文人的缺点——“心比天高，命比纸薄”，什么也做不了，愤世嫉俗。但是如果只是行动者，那就不可能成为杰出的工程工作人员。所以作为架构师，我觉得应该二者兼备。

《程序员》：从您这些年来的观察，您觉得现在我们的软件业界最缺思考者还是行动者？

李伟：咱们应该说不缺思考者，但是很不幸的是：由于种种客观条件的局限，造成了咱们的思考者没有在架构方向上健康成长，这是要命的一点。如果中国的架构从业人员，能够被10个世界级的、架构方面的杰出人士引领，那是我的梦想。那样你才能正确思考。否则，没有这样的引领，思考就会有很大的局限性。至于行动者，我们当中就有非常杰出和优秀的技术和研究人员，尤其是一些高级技术或研究人员，他们的水平是很高的，是出类拔萃的，而且不能单纯地说他们就比架构师差，这完全是一种错误的观点。我个人认为：研究算法的一些人员的水平，一点都不比高水平的架构师差，只是工作的方向不一样。

打造个人核心价值

《程序员》：说起德国企业，有人觉得不够灵活，只知道按规矩办事。那么您在西门子中国研究院工作这么长时间，是什么样的感觉？对这一点有何体会？

李伟：对于这个说法，我觉得也可以用另外一个视角看。我可以把它形容成很工程化、很严谨、很守规矩、很注重纪律。德国这个国家和德国的企业，身上烙下了德国的影子，就是“工匠”。他们很多人以自己为工程师、或者是高级工匠专业技术人员而骄傲，而不是觉得可悲。他们还经常觉得一些玩儿商业的人没有社会价值。他们为自己是一个鞋匠而骄傲，而并没有觉得开鞋店的老板应该值得羡慕。

这点和中国人想法不一样，首先是国家文化的区别。另外一个区别就是：德国既往的历史告诉他们一个很重要的历史经验——做任何一件事情，无论是做工程化的东西，还是做一双鞋，还是进行战争，如果你遵循了系统化的思维习惯、一板一眼的工作作风、严格按照规矩去办事、形成团队作战，并且要讲求纪律来服从最高的领导，就可以形成很强的战斗力。这是德国人都知道的历史，所以他们深信：只有这样做，集团作战，才能把一件事做成功。他们觉得不这样做就没有第二种方式。“怎么可能做事不遵循最高原则、没有纪律，做事不系统化、全方位地思考？然后做事不按照既往的、别人成功的这个做法严格执行？应该在这基础之上进行创新，而不要一味的追求创新，标新立异。

他们想不到第二种做法，同时我个人也想不到第二种做法。我不认为不借鉴前人的经验、严格遵循纪律，就能做得成功，我不相信。第二，我也不相信不遵循一些系统化的做事方法，就能把事做得精彩。纪律，在很大的程度上是高于个人利益或者是团队利益。没有规矩不成方圆，做事肯定要遵循纪律，要遵循次序，这就是我对德国人的理解。

当然我这好像是在谈德国人的工匠风格如何好，倒不是这个意思。我想任何事情都是双面的，我们需要吸收双面的优点。

《程序员》：国内的软件开发人员都有一种焦虑感，觉得自己在技术上30岁之后很难发展，倾向于去做管理，您对此有何建议？

李伟：我比较注重于个人价值的构建工作，这意味着：我会想现在需要会些什么东西，能够给公司和社会带来什么样的价值；未来打算在哪些方面给公司和社会提供我个人的什么价值。我信奉一个原则：只要你在社会当中有价值，能够创造价值（不是捏造自己的价值），你就会获得社会的回报。

我觉得作为中国人，尤其是作为中国年轻人，还是应该更多去想我说的那方面的思维习惯：价值是能给你带来社会回报的。如果一味只追求形式，人一辈子经常起起伏伏，你没有了价值，即便是你现在做了某个大公司的CEO，也可能有朝一日成为社会底层的人，这在美国已经出现了。

当然这个话说得有点空，但是我只是在强调：无论你走上哪个行业，即便是技术还是管理都没关系，但是要注重的是个人核心价值，要时刻提醒我有什么价值，我能给我的团体带来什么，我们能给我的社会带来什么。

尤金·卡巴斯基解读2009年信息安全趋势

——将网络罪犯关进“恶魔岛”

■ 记者 / 付江

尤金·卡巴斯基(卡巴斯基实验室创始人兼CEO)十天的“2009安全中国行”之旅于4月2日启动，安排相当紧凑，“卡饭”见面会、媒体宣讲、旗舰店签售、最高学府演讲，足迹遍历北京、上海、广州。作为世界顶尖级的电脑病毒专家，卡巴斯基与大家分享着信息安全行业的最新研究成果。

2009信息安全形势将持续恶化

在卡巴斯基看来，今年全球信息安全形势还将持续恶化。最重要的原因就是网络犯罪从之前的单纯破坏、炫耀到如今的金钱利益，整个产业链已经成熟。“时下的网络犯罪组织分工明确，从探测漏洞程序、制作木马到提供僵尸网络服务，根据提供的天数、技术难度都有明确的价格，由一系列专门公司来提供相应技术支持和服务。”在谈到金融危机带来的影响时，卡巴斯基认为，这将导致网络犯罪数量成倍增长，经济危机对于业界计算机高手来说可能意味着失业，因此他们有更多自由时间，却没有钱，这或将导致他们铤而走险迈向犯罪道路，比如盗取信用卡密码、盗取网游账号、装备卖钱等网络犯罪，这已经呈现出一种激增态势。

恶意病毒木马将广泛入侵智能设备

未来操作系统种类会比现在多的多，更友好更灵活，但基于种类繁多的操作系统的恶意程序也会随之越来越多，近年基于Mac、Linux的病毒都呈逐年上涨态势也可说明这一点。不仅如此，未来智能手机上的应用软件会极大丰富，功能比如今单纯的计算机更强大，会有越来越多的人编写针对移动设备操作的恶意软件。而随着智能家庭时代的逐渐走近，如今运行WindowsXP的咖啡机已经出现，还有与网络连接的洗衣机、洗碗机等各种各样的智能家用电器，所以会有人通过互联网肆意操作它们，或者利用它们来传播恶意软件或垃圾邮件，这些威胁理念现在都已经得到了证实。

鉴于网络犯罪越来越有组织性，卡巴斯基甚至设想建立“互联网国际刑警组织”来打击跨国网络犯罪，这个机构需要24小时不间断实时监控网络，打击那些无边界并在网络上大肆横行的犯罪组织。对于如何处置网络犯罪分子，卡巴斯基笑言，应该对他们限制适当的人身自由，把他们都关在关门的监狱里，监狱的名字叫“网络恶魔岛”(恶魔岛是美国历史上防范最严密的监狱)。

云安全模式与众不同

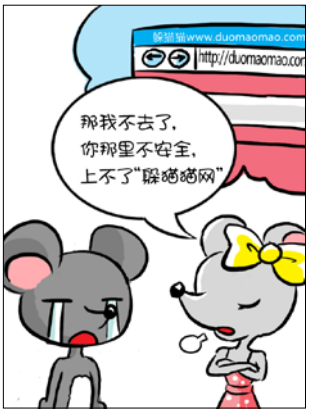
对于卡巴斯基2010版，全面引入“沙盒”技术是其最大特色之一。杀毒软件将对那些未知的、尚不能判断出是否恶意的、不明来历的威胁，通通装入“沙盒”这一相对隔离的环境中运行，实时监控，如果出现异常情况就将其在对系统毫无影响的情况下彻底删除。

卡巴斯基实验室创始人兼CEO尤金·卡巴斯基

而关于当前热门的“云安全”技术，卡巴斯基亚太区技术总监王南告诉《程序员》杂志记者，卡巴斯基的模式与现在流行的两种传统模式不同(即瑞星的将广大互联网用户作为“云”端和趋势科技的以在服务器集群建立黑白名单库作为“云”端)，卡巴斯基采取的是在整合两种方案基础上的独特模式，而关键的数据控制权则完全由用户掌握，用户可以自由选择是否将即时的潜在木马威胁提供给卡巴斯基数据库实现分析共享。

展望2009年，即使是在全球经济下滑的今天，卡巴斯基还是对今年的业务发展充满信心。2008年已在中国设立了全球产品测试中心，作为除莫斯科总部之外最大的产品研发中心，卡巴斯基公司还在中国8个城市设立了技术支持中心。卡巴斯基透露，2009年会持续加大对中国市场的投入，将会在中国招募软件工程师，进一步推动产品在中国市场的本地化进程。■

躲猫猫网



幽默

茶杯架

某日，某公司新买一台微机。公司经理乃一位好事之人，虽已年近半百，仍对计算机表现出浓厚的兴趣，整日端坐于前，乐此不彼。

一周后，电脑公司收到该经理的电话："你是xx电脑公司吗？我是xx公司，我们上星期刚从你们那里买了一台电脑，可是到今天，电脑上的茶杯架就坏了。这是咋回事？"

"哪里有电脑上茶杯架？"

"就是电脑上的那个一摁就出来的放茶杯的那个东西呀……"

初学者

一日，机房里两个电脑初学者正用Word忙着做什么东西。突然初学者甲说：

"你打错了一个字。"

乙依言一查，果然如此，问甲："怎么办？"

甲答曰："这有什么难的？关机，重启！"

……

看电视

老婆正在看电视，突然没台了。

老婆：怎么没台了？老公检查一下！

老公：哦！蓝屏了。

老婆：为什么呀？

老公：不是病毒发作就是硬件冲突。

电脑与饮酒

电脑公司开业之际，亲朋好友饮酒助兴。

一声"开机(启)"，大家各自开启一瓶啤酒。

"清零"，大家举瓶畅饮，进行一次"批处理"。

"复位"，放下酒瓶。

……

"嘿，别喝了，我的内存不够，没法运行。"甲拍拍肚皮道。

"可不，我的显示器也出毛病了。"乙颤抖着手，语无伦次地说。

"哎，我的键盘怎么失灵了。"丙叫嚷，"眼前一切都飘飘摇摇的。"

电脑和人脑

一天，一个人和一台电脑比谁的能耐大。

电脑说："现在是我的世界，各行各业都要用我，人类没我不行！"

这时，这个人走过来，把电脑关了。

声音

"现在，我们正经历一场变革，一场技术与平台的变革。这场变革，可能是有史以来最重要的。"

——CSDN&《程序员》总裁蒋涛在2009中国软件技术英雄会（北京站）上表示

"简约Java时代已经开始了！"

——Spring之父Rod Johnson在QCon2009全球企业开发大会（北京站）上表示

■ 策划 / 本刊编辑部
■ 文 / 周至

移动应用修炼之道

IT界的移动领域，如今暗潮涌动。3G、AppStore、Android、iPhone，甚至是山寨，一个个接踵而来的名词不断地回响在我们的耳边，经过这个移动终端、网络、应用、平台和内容各大运营商争夺用户的“乱世”之后，新一代移动平台及其生态系统即将诞生了。

而最能反映整个移动生态链变化的，便是基于移动应用的诸多产品。从它们的创意产生，直至捞到第一桶金，每一个环节都与整个生态链息息相关。而且，“得应用者得天下”一句并不为过。拥有一个好的移动应用产品，不仅能够使厂商广为获利，更重要的是同时能够虏获千万用户的心。

本期特别策划，就将围绕移动应用产品的生命周期，将移动产品最有特点的生产环节进行拆分、组合，为读者讲述移动应用产品的修炼之道，同时反映出新一代移动平台的生态状况，在更高的层次上关注移动产品，关注移动领域。

整个特别策划的开篇，在纵览移动生态风云变化之后，深入地介绍了Opera Mini这款知名移动浏览器产品的生产过程及发展史；中篇则将一些具有特点的产品生产过程进行细分，选取各自最有特点的部分进行介绍，虽分篇展示，但环环相扣，具体内容涵盖：GameLoft最受欢迎游戏的创意揭秘、Android Market应用开发者的开发经验分享以及移动应用的部署实践和安全处理等。

全篇特别策划，以移动创业者的创业故事作为收尾，并在最后为读者介绍了诸多富有特点的移动应用，让读者在阅读之后，晓尽移动产品的那些事儿。

移动生态风云变

■ 文 / 马宁

2009年，移动领域的生态风云大变，智能手机和移动应用的概念已经深入人心，3G牌照的发放为我们创造了更多的机会。作为一个移动开发者，身处其中，我们是幸运的。回顾以往的产业革新，最早的领跑者并不一定会是最后的胜利者。究其原因，在产业快速上升期，领跑者往往会忽略对于未来的预见与把握。所以，我们在3G的前夜，试图分析整个移动应用领域生态链的变化，以及未来的趋势。

3G与丰饶经济学

3G是什么？3G是一条高速公路，将在线服务提供者与终端用户连接起来，并且将物流成本下降至接近零。这符合“长尾理论”中丰饶经济学的描述，当仓储、物流成本接近零时，为用户提供尽可能多的服务将成为业务增长的主要模式。这也是iPhone的AppStore模式受到青睐的理论基础。

受3G影响最大的是运营商。2G时代，运营商是唯一的主宰，掌握面向最终用户的入口资源，提供通信、带宽、内容等所有的服务。但随着3G时代的到来，传输带宽将无限增加，由运营商提供的内容将无法满足用户的个性化需求。运营商将转而成为3G时代的管理者、推动者和服务者。虽然运营商仍将占据主流地位，但是采用了标准网络协议的3G时代，运营商将无法通过封闭的移动网络来将竞争者挡在门外。一个开放的移动互联网将在3G时代逐渐成型。

操作系统之争

由于国外的3G比我们要早两三年的时间，最近手机操作系统的火热也与3G的大背景密不可分。手机厂商和运营商只能提供有限的应用和服务，而更多的服务则需要由ISV和互联网门户来提供。

手机操作系统一直都是手机厂商的天下，而由在线服务商Google推出的Android则改变了这种格局。在线服务与操作系统之间的联系越来越紧密，在操作系统中植入在线服务，将成为一种新的盈利模式。

3G时代，对于手机操作系统的要求莫过于开放性和标准化。手机操作系统的SDK也直接决定了手机上应用的数量。iPhone的AppStore需要有SDK的支持，才能达到预期效果。将操作系统提供的服务，比如电话、媒体播放、网络等，封装到SDK中，让开发者也可以方便的使用这些系统级服务。

客户端

移动应用的开发也不仅限于传统应用程序开发领域，.NET CF、Java ME和C++的争夺已经不那么重要了，更为敏捷、轻量级的开发技术已经出现——基于网页开发技术的移动应用。Flash Lite、Silverlight Mobile等RIA技术不存在多平台兼容的问题，并已经显露出巨大的潜力。JavaScript、AJAX等技术受到网络间断在线的限制，在移动平台上的应用受到限制。不过随着Google开发的CodeGear和HTML 5等技术的发展，技术的缺陷终将被弥合。

Widgate的潜力也不可忽视。用C++、.NET和Java为不同的手机开发应用，将成为在线服务提供商的噩梦，尤其是那些实力弱小的网站。而这些网站最熟悉的开发技术是什么?HTML、JavaScript。恰好，Widgate所需的开发技术也是这些。将已有的网站略加修改，就可以为不同的手机提供在线服务，这将吸引更多互联网公司进入移动领域。

抢占制高点—云计算

我们已经多次提到了在线服务，在线服务从哪里来？2G时代，服务来自运营商的服务供应商（SP）。开放互联网的精神已经深入人心，依赖少数SP一统天下的想法恐怕过时了。正确答案是，在线服务来自云端。任何信息都可能成为服务的内容，从卫星照片到租房信息，这些内容将在被重新整合后，以某种统一的形式推送给终端用户，或者由用户通过标准的接口进行查询。这就是云计算的思想。

AppStore就是移动服务的初级模式，它提供的是移动应用的在线下载服务。虽然现在AppStore模式甚嚣尘上，不过也许数年后，大部分的AppStore也会归于沉寂。原因在于，内容的数量不足以支撑这种盈利模式的长期发展。而且，AppStore也没有重新利用移动设备的特点，将服务内容进行重新的整合。

那么，移动服务的高级模式是什么？至今还不能预见。移动设备具有随身携带的特点，所以，在线服务影响现实生活，也许会是移动服务发展的新途径，比如电子优惠卷、移动购物等。但移动服务的模式一定是大公司提供平台，小公司或个人提供服务。Marketplace模式将深入人心，而卖的东西也许会超出我们的想象。■

一个手机浏览器的发展史

■ 文 / Zi Bin Cheah　高伟

2005年四月，挪威首都奥斯陆北欧的春天才刚刚到来，地面还覆盖着零星白雪。

在Opera总部5层的办公楼里，Opera软件公司全球CEO谭咏文（Jon S. von Tetzchner），一个身高足有2米而看起来有些严肃的冰岛人，倚坐在沙发扶手上，仔细摆弄着手里的一款普通手机。一切源于公司瑞典研发部门里几个年轻工程师开发的一个小程序。

利用这个不足100K大小的程序，谭咏文在一款非常低端的手机上轻松地实现了快捷浏览互联网。

作为公司的CEO和网络浏览器产业的先驱者之一，谭咏文敏锐地看到了这个小程序了不起的未来。

这个小程序就是后来风靡全球的Opera Mini移动互联网浏览器。作为全球最受欢迎的手机浏览器，Opera Mini拥有超过2000万的活跃用户。自2006年发布至今，Opera Mini累计下载量已经超过了4亿次。而根据Opera 2009年2月公布的官方统计数字，Opera Mini用户每月浏览的数据量高达1PB，这大约相当于全球最大的社交网站Facebook上存储的所有照片体积的总和（约100亿张图片）！

总部位于挪威首都奥斯陆的Opera软件公司是一家老牌的浏览器开发公司，其开发出世界上最早的一批互联网浏览器之一。多年来，Opera一直心无旁骛，完全专注在浏览器的开发上。尽管由于受到微软在Windows捆绑IE等原因影响，市场份额一直受到挤压，Opera重视技术创新的特点却让这家不大的北欧公司历经多年风雨而屹立不倒。

不仅如此，公司还把目光放向了比PC更广阔的移动和其他设备市场。

“Opera的愿景一直是One Web（一个互联网），我们致力于帮助用户实现无论何时何地通过何种终端都能拥有完美浏览互联网的体验。”谭咏文说。在Opera，每一个员工的名片上都印着这样一行字：Best Internet Experience On Any Device。

Opera Mini的诞生

秉承One Web的理念，早在2000年，Opera首先开发出世界上最早的一款能在手机上实现完整互联网浏览体验的浏览器Opera Mobile。经过多年发展，这款具有划时代意义的移动浏览器已经具有支持多窗口、页面可缩放、支持触摸操作、文件下载、浏览信息同步等强大功能，用户几乎可以在手机上实现PC浏览的所有体验。

无疑，Opera Mobile的设计和运作都是非常成功的。很快，Opera公司就攻下了中高端手机，包括智能手机这块在当时被普遍认为发展潜力很大的市场。

迄今为止，Opera Mobile也是世界上出货量最多的一款全WWW手机浏览器，被各大手机厂商所追捧。据不完全统计，仅2006年至今，预装Opera Mobile的手机出货量就已超过2亿台。

然而在2005年的时候，无论是对于致力于推广One Web的谭咏文自己，还是对于当时Opera公司业绩发展而言，Opera有更大的想法。因为，即使拥有中高端手机市场的庞大的市场份额，在全球范围内，还是有相当一部分用户，不管是由于网络速度的原因，或者是低端、超低端手机的限制，不能够最流畅地访问WWW网页。

“从营收角度讲，Opera当时大可以继续把注意力放在Opera Mobile的发展上，进而扩大自己高端品牌的形象，但Opera的愿景是希望所有人，不论国家和地区，在所有设备上都能够自由地访问互联网，不管是一台老旧的PC，还是一款陈年的手机。Opera人对互联网能够被所有人使用，有很强的使命感。”当时还在奥斯陆总部工作的Opera中国区总经理宋麟回忆说。

这是一个很困难又很容易的决定。当时Opera公司的资源非常有限，研发人力全力投入到下一代浏览器的研发当中，分出人力来做Opera Mini不是很轻松，然而，公司认为，为了照顾同样多数的中低端手机以及普遍速率不快的2G网络，Opera应该主动把WWW互联网带到所有的设备和网络上，而不是等待WAP的消亡和高速网络的全面普及。

最终，一个一致达成的结果是：

在继续开发下一代顶尖浏览器的同时，开发一款能在非智能手机上运行的Opera Mobile简化版本。这款新浏览器应该能给手机增加最少的负荷，必须能够在多数的中端和低端手机运行，可以在最基本的2G网络下也能有最好的表现，至于新浏览器的名称就叫做Opera Mini，即迷你版的Opera Mobile。

Opera Mini内置Google搜索引擎

Opera Mini的设计要求

在决定开发Opera Mini之后，Opera公司提出了这款产品的四个基本设计要求，即：

1. 能够运行在绝大多数的手机上；

2. 体积小巧；

3. 速度快捷；

4. 能显示绝大多数的网站。

既然是为广大市场开发的产品，那么首先Mini就必须能运行在绝大多数的手机上。然而，中低端手机种类繁多，数量庞杂，Opera无法根据不同手机机型逐个进行开发，而且这也不利于后期升级和维护。最好能找到一个共同点做平台开发——Java正好符合这个要求。公司的研发团队经过慎重讨论最后决定，Opera Mini将会是一个J2ME软件。

在选择了J2ME后，第二个遇到的问题是软件体积问题。中低端手机的内存有限——当时的手机有个几兆的内存已经是神奇了——如果文件过大，手机就无法安装。即使可以勉强装得下，许多中低端手机也没办法像高端手机那样能够执行浏览器的图片渲染、执行脚本等功能。于是，程序员的另一个目标就是：Mini必须体积小巧的同时也可以执行浏览。经过研究，研发团队决定做一个瘦客户端的程序，让服务器群组来承担网页的解析。传送到用户的Mini里的将是一个优化之后的页面，这就是Opera专门开发的OBML（Opera Binary Markup Language）页面。

解决了软件体积问题，下一个问题就是如何提高用户浏览速度的问题。这是考虑到两方面原因：第一，流量产生的庞大通信费用问题；其次，中低端手机本身内存有限。这样，除了对页面的优化，研发团队还决定要对页面本身进行压缩。压缩之后的OBML页面大小能小到原来的10%。这样就大大地减少了手机的负担，同时也降低了流量，从而提高速度，并节省通信费用。

最后，既然在手机上运行，Mini必须能够支持尽可能多的网站。由于历史原因，在手机刚刚支持浏览的时候，业界开发了特别针对手机页面的技术WAP。许多网站开发了WAP页面，同时也有许多WAP网站直接上线。然而Opera一直的理念是One Web，Opera希望能够最大力度支持互联网；所以在完全支持旧有WAP的同时，新的浏览器能够最大限度在受限的设备网络条件下完美显示WWW网页。好在研发团队本来就决定了使用瘦客户端-服务器群组架构，这样的好处就是Opera服务器将承担处理不同技术的页面，而最后到达用户手机里，只会是一个适应当时手机屏幕的完美呈现。所有问题全部解决。

Opera Mini成长之路

设计确定后，Mini研发工作理所当然的由最初提出创意的瑞典年轻的研发团队承担。

Opera瑞典的办公室位于瑞典中部安静的小城Linkoping。大部分工程师来自附近著名的Linkoping大学。办公室坐落在一栋19世纪的老式楼房中，地面铺着地毯，屋顶缀着吊灯。开发人员每周打桥牌，参加乒乓球“联赛”，下了班大都踱着步慢慢遛弯儿穿过绿

树成荫的小街回家。

“整个团队有一种安静而神秘的气氛，很少被打扰，在Emacs里使用Java，C/C++以及很少人用过的Pike语言进行开发。团队成员使用一种叫Lyskom的系统交流，它结合了E-mail、即时通信和会议系统等元素，工作之余还会在上面聊一些好玩的话题，比如怎样黑掉公司作为圣诞礼物送给大家的电子相框，哪个ERP系统最烂，以及乐高最新的可编程机器人。”当时曾在瑞典参与开发的中国团队工程师冯嘉彬说。

开发工作在瑞典小城富有创造力的环境中飞速进展。

2005年8月，Opera Mini的实验版首先在挪威进行了小规模的推广。结果反响热烈。

同年10月，Opera Mini的beta版本在瑞典、丹麦、挪威和芬兰四国进行了试用，迅速获得用户的好评。

同年11月，Opera Mini的正式版本在德国推出。12月，Opera在自己的官方主页上悄然放置了Opera Mini下载链接。

2006年1月24日，Opera Mini正式向全球发布。

5月，Opera Mini 2.0上线。增加了文件下载、可定制皮肤、可选搜索引擎、快速拨号等。

11月，Opera Mini 3 beta发布。增加了浏览安全控制、RSS支持、图片上传支持等。

一年后，2007年11月，Opera Mini 4震撼上市。根据市场反馈和公司一贯的创新特质，Opera重写了Mini的所有代码。全新的Opera Mini 4首次引入了概览图和页面缩放功能，使得用户在浏览互联网网页时能看到类似PC浏览器上所显示的页面格局，而不用去适应由于手机对页面格局进行调整而带来的陌生感。不仅如此，Opera Mini 4还支持Opera Link（浏览同步）。这样，用户可以将日常浏览的书签、快速拨号、浏览历史等个人信息与其PC浏览器保持同步，带来移动便利。

2007年7月19日，Opera同国内合作伙伴空中网联手，发布空中Opera Mini，将Opera Mini引入中国，而2008年12月16日，Opera推出Opera Mini中国版，版本是4.2。在中国架设了服务器之后，中国用户也能利用手机轻松访问互联网。

Opera Mini的盈利模式

一个好的产品不仅在技术上需要有好的创意，还需要能在市场上存活和盈利。对于Mini，Opera已经有了成熟的盈利模式。

“Opera Mini目前的盈利模式主要有两种，即B2B和B2C。”Opera中国区总经理宋麟指出。

就B2B而言，Mini主要是通过预装销售的模式获得利润。先进的技术和成熟的产品，使得Opera Mini被众多手机生产厂商和移动运营商所青睐。2009年2月，Opera宣布与全球最大的移动运营商Vodafone签署协议，Vodafone全面预装Opera Mini。Opera的合作电信运营巨头还有美国的Virgin Mobile和T-Mobile、欧洲的Telefonica、法国的Ten by Orange、日本的KDDI等。选择预装Opera Mini的手机制造龙头则有诺基亚、三星等。

就B2C而言，Mini采取的方式是流量分成。例如，Mini目前预置的搜索引擎是Google，根据Opera和Google的协议，手机用户每一次在Mini搜索框中搜索时，Opera都可以从中抽取微量利润。

“当然，对个人终端用户而言，Mini是免费的，用户可自由下载安装使用而不需要花一分钱。”宋麟补充说。

随着技术和市场的发展，还会有其他的盈利模式出现，宋麟表示。

Opera Mini的未来

Opera Mini曾被误解为一个过渡期的产品。很多人认为，Opera公司开发Mini的主要目的是为了在手机市场从非智能手机主导转变为智能手机主导之前的这段时期里，能进入到巨大的非智能手机市场。

从这个角度上来讲，Opera Mini的价值似乎将随着智能手机的发展而逐步减少。

然而，时至今日，Opera Mini现在全球的活跃用户已经超过两千万，而用户增长的势头仍然迅猛，年增长率超过70%。Opera Mini用户每月流量用量已经超过1PB。

不仅如此，与当初设计初衷不同的是，也有相当多的中高端智能手机用户选择使用Opera Mini，因为它的速度优势，流量的节省和一流的网页展现。

“我们对Opera Mini的前景非常看好。”谭咏文表示。

“在中国，虽然3G牌照已经发放，但3G网络的建设和成熟，以及手机的支持还需要大约2年的时间，Opera Mini虽然进入中国的时间还不长，但希望在以后的时间里，将同Opera Mobile以及中国移动互联网业界的同仁一道，为中国用户在手机上拥有最完美的浏览WWW网页体验做出自己的贡献。”Opera中国区总经理宋麟说。

除了飞速发展的中国和东亚近邻，欧洲和美国，在很多发展中国家，Mini的表现也是惊人的。近年来，印尼等东南亚国家，包括很多非洲国家使用Mini上网的人激增。

“我们发现有很多用户是第一次上网——他们之前并没有用过计算机。这就打破了一个固有思维：以前人们都会觉得手机上网通常会由那些技术派的人带动，可是在发展中国家，我们却发现了一群第一次上网的用户。”马来西亚华裔网络标准专家谢子斌说，“从这个意义上讲，Opera Mini将承担更多的历史使命。”

带玩家进入游戏性之环

——Gameloft“午夜保龄”创意全揭密

■ 余非

对于一款移动应用产品来说，尤其是手机游戏，创意都将处在绝对的核心位置。因为，技术始终还是要为人的想法来服务的。开发一款游戏，游戏策划人员首先要有一个新鲜的想法，然后自始至终地将这个想法作为游戏的生命力的最主要部分看待。只有这个部分有了，其他的相关技术才能围绕其服务，从而打造出一个优秀的手机游戏产品。

这个过程就好像你去看一部动画片，比如皮克斯的WALL-E，或是梦工厂的功夫PANDA。从他们的剧组采访中就可以看出，对于一个3D影片、动画片，最重要的就是一个感人、吸引人的好故事，并且用一个很好的方式把这个故事讲清楚、讲明白。其次才是利用各种技术去辅助将其做好。同样，想法对于技术也会起到一个促进的作用，就是因为你有了很多想法，才会想到更多的技术手段去实现这些想法，即突进式的想法才会促使突进式的技术产生。

接下来，我会围绕一款由Gameloft（智乐软件）有限公司倾力打造的，在AppStore上热卖的手机游戏产品——午夜保龄（Midnight Bowling），来讲述一个好的游戏产品创意是如何诞生的。

创意火花的产生

午夜保龄，是由Gameloft开发的运动类产品线与午夜系列相结合的一款游戏，其他同类型的产品还有东京之夜、巴黎之夜等。

在构思整个游戏的创意之前，游戏策划人员应该首先确定这个游戏所面对的人群。比如午夜保龄这款游戏，它的策划者就首先将游戏的受众确定为年轻人这个当前的主要消费群体。午夜，是大多数年轻人都比较兴奋的时候，这个时间可能很多人会去跳舞，然后去夜店喝酒，和朋友一块聊天，做一些比较酷的事情，包括谈恋爱，玩一些他们比较喜欢的运动等，整体感觉会比较Fashion。

而午夜保龄的“午夜”，就会让玩这款游戏的年轻人感觉到他们身处一个时尚的大都市，然后很多人去做一些比较酷的、流行的，年轻人喜欢的事情，而没有那么多条条框框，甚至有一些地下的感觉。然后，“保龄”又是一个大众化的运动，无论是专业还是普通的玩家都比较喜欢玩，操作也不复杂，有很多用户在实际中都玩过。所以，综合这两点，符合这个创意的游戏就会吸引一个很大的玩家用户群。

此外，我们还希望玩家能有更好的体验，游戏的场地拥有世界各地的不同风格，有夏威夷的感觉，中国香港的感觉，东京的感觉等。玩家所扮演角色的周围还会有一些热闹的人群，有时玩家也可以看到吧台甚至和他一起游戏的其他玩家，包括对手和伙伴。他们可以是比较流行的男孩和女孩，也可能是比较帅的或有个性的成熟男性和女性。每个角色的服饰、人物性格甚至口音都不太一样。在午夜保龄这款游戏里，为了做到每个人物说话的口音都不相同，游戏开发人员找了世界各地区的人为角色分别进行了配音。

“午夜保龄”游戏中富有个性的人物角色

让玩家钻进这个环

在游戏策划的初期，能够有一个好的创意，并且把游戏的本质做好，就能制作出一款好的游戏，并且吸引到大量的玩家用户群体。但是一个真正优秀的创意，不仅在于能够做好一款游戏，还要考虑这个游戏具有多长的生命力。无论任何一个玩家，他购买任何一款游戏时，都希望游戏对他来说能够有更久的吸引乐趣，这样他玩起来才会尽兴。那么，我们就希望设计这样一个环，让玩家钻进去，乐不思蜀。

第一步，让玩家有渴望的感觉。我们小时候，尤其是男孩子，可能都曾很想要一个滑板或自行车。在这个过程中，你肯定是先看到人家骑自行

车，觉得特别有趣，才会产生渴望的感觉。这就好像玩家看到“午夜保龄”这款游戏一样，要让他觉得真好玩，画面真漂亮，人物真帅，打起来感觉特别兴奋，才会产生渴望的感觉，进而想办法得到它，这是人的一个本能。

当玩家产生了对游戏的渴望，就到了策划者应该考虑的第二步，让玩家在游戏中体会经过努力才能得到的一个过程。这个过程的设计很重要，就好像我们通常会说的，有钱人家的孩子对很多东西都没什么兴趣了，给他名车，或者漂亮姑娘、帅小伙子在他身边，他都不太在意，因为这些对他来说太容易得到了。所以，在“午夜保龄”这款游戏中，我们要设计那么一个过程，无论是一个新人物、新的保龄球还是一个新的技能，都要通过一番努力和思考，才能够获得。这就好像你小时候想要一款游戏机，攒了一年的钱，然后买了这款游戏机，你这个时候拿到它，才会觉得特别的兴奋，这也是人的本能。

第三步，让玩家体会学习的过程。当玩家得到一个新人物之后，就要让他新学一项技能，比如一种新的打球技巧。学习是很大的一个乐趣，尤其对于运动类的游戏，比如保龄球。虽然这之中也许会有挫折，但是经过玩家不断地进步，最终学会了，这对于他来说将是非常大的乐趣。

接下来是第四步，给玩家一个机会让他去应用新学到的东西。比如，你学会很帅气地调酒，给别人展示你娴熟的调酒技巧，并且吸引到了你感兴趣的人，让一些比较漂亮的女孩或男孩夸你酷，你就会很享受这件事情。但如果没有这个过程，当你学好一件事，还没有机会去显摆你的技能，就继续进入了另一个痛苦的学习过程，这样玩家很快就会失去信心和乐趣。

在玩家足够地享受了“炫耀”的过程之后，就要让他将这个能力再去强化，在“午夜保龄”中，当你学会了打保龄球之后，你可以选择不同的球使用加强的技能，或者是不同的人到不同的地区、跟不同Level的挑战者来挑战，这就是第五步了。

我认为，这样的5个环节就成为了牢牢套住玩家的一个圈，当玩家完整地享受过这个过程之后，你要马上让他进入这个循环的开始处，让他产生对另一种能力的渴望，这样玩家就会对游戏产生极大的粘性。当然，在玩家学习到很多不同的能力以后，要考虑让他掌握能力的混用，在混搭的过程中创造更多的乐趣。

努力将创意实现

很多游戏策划人员可能会考虑到如何在技术层面上将创意实现的问题。我认为，从技术角度上，什么都是有可能的。如果抛开平台的限制不谈，任何想法都有可能找到途径去实现，只是有些实现会很难，有些相对来说简单而已。如果考虑平台与技术力的限制，例如iPhone的性能不能跟最高端的计算机来比，但是我们依然可以用折中的想法或换一个角度做出类似的效果来。

举个例子来讲，比如你想建立一个游戏世界，里面有1000个人，那么在技术上可能实现不了1000个人同时显示，也许只能显示100个人。但我们可以换一种思维，同屏内只显示100个人，当玩家走到另一个区域，再创建其他的人。最后，一共还是一千个不同的人，只是换了一种技术上要求没有那么高的办法来实现而已。

针对“午夜保龄”这款游戏，因为它并没有涉及到像RPG或像足球一样要同时显示很多人，也不会像战争游戏那样有大量的战争机器，所以在显示方面的问题可能不那么突显。但是，它会遇到的问题在于iPhone有限的3D图形处理能力——同时处理的多边形数量是有限的。那么，场地的实际开发就会和原来的创意有所矛盾：如果你想把所有东西做的很漂亮，这样这个东西就展不开；或者是想把很多东西加进来，又不会很漂亮，会把多边形分散的很厉害。所以，我们必须要把有限的资源和有限的技能去放在最重要、最关心的部分——把重点放在游戏人物的操纵上和球的身上。

在这两个部分做完了以后，就要考虑如何能用多媒体做出尽可能好的场景，并且有所重点。我们会尽可能地把多边形集中在球道等最出效果的地方。比如，你觉得一个吧台上什么东西会给玩家印象最深，一个壁画还是一个酒瓶？那么就要突出它，这就是注重实现创意的细节。又比如一个保龄球场地，如果在夏威夷，玩家也许会希望突出的部分是有夏威夷特色的，而有没有沙发则无所谓，那么就完全可以把沙发去掉。对于人物也一样，在做人物多边形的部分时，远一点地方的观众可以就做成片状的，但实时操作的人物，肯定就要使用3D的角色。充分利用现有资源，尽可能地实现你的创意，这一点非常重要。

瞄准投球阶段

这里玩家可以用手指按下方两个星星的区域来调整瞄准线的左右位置。

而如果按下保龄球上的星星，会进入选球界面。

投球阶段

这里完全模拟了保龄球的投球动作，水平横向握住手机，先下后上，象真正扔保龄球一样完成动作，这时游戏中的人物就会在手机静止后1秒，将球投出。

寻找金矿的入口

■ 文 / 胡笛　余宙

新潮流的流行和兴起，正如《引爆点》所探讨的一样，没人知道具体的传播路径，也没人知道具体的爆发时间。只觉得“嘭”地一声，似乎一夜之间全世界就开始沸腾了。对于个人开发者来说，移动设备上的开发也似乎如此。在此之前，大家只知道唯有强大实力的内容提供商才可能绑上运营商的大船，跟着一起乘风破浪，大发其财。个人开发者只能望洋兴叹，空有聪明才智，却苦无门路让自己的程序搭载到每台手机上。

然而，在2007年1月9日，苹果正式推出了iPhone。正如《引爆点》里的传说一样，iPhone也是“嘭”地一下，就开始流行起来了。被世人所称道的，不仅仅是iPhone创新无比的外形、华丽的用户界面、独特的操作方式。而更多的，是iPhone的AppStore为用户带来的财富传奇。

在最近的传闻中，AppStore新的传奇缔造者是Ethan Nicholas，他的iShoot程序，在2009.1.3日发布，十天之后就以近1.7万的下载量飙升至付费软件的第一名，扣除苹果的运营费用，每天的收入多达2.1万美元。而目前的排名第一的软件，下载量更是高达百万。而苹果上总计的下载量更是即将突破10亿。这怎能不让人眼热心跳，这怎能不让各大公司上蹿下跳？

赚钱的事情，总不会只有一个玩家。而今天，我们要探讨的就是在移动平台上声势正隆、采用完全开放的姿态、拥有同样的游戏规则的另外一个平台——Android。

Android的缘起

翻开历史的数据来看，不得不让人佩服Google的远见卓识。在2005年7月，Google收购了Android,Inc.，一个在美国加州的小型创业公司。在当时传出的消息，仅仅称他们是家做移动设备上软件的公司。2006年12月，来自BBC和华尔街日报的报道提及了Google希望在移动手机上提供搜索和程序服务，人们更多地在猜测Google的确想在移动市场上有所作为。2007年9月，InformationWeek公布，Google申请了移动领域相关的一些专利。2007年11月，Google号召建立了开放手机联盟，这个联盟阵容强大，包括Google、HTC(宏达电)、T-Mobile、高通、摩托罗拉、三星、LG以及中国移动在内的34家企业都将基于该平台开发手机的新型业务。

这时，世人才真正看清Google的庞大计划。谁能料想，两年的时间，Google就折腾出了一套手机操作系统！2007年11月5日，Google正式宣布Android为其基于Linux平台开源手机操作系统的名称，该平台由操作系统、中间件、用户界面和应用软件组成，号称是首个为移动终端打造的真正开放和完整的移动软件。这个时候，苹果AppStore的成功已经让大家看到了手机设备商撬动移动运营商的机会和可能，以及个人在移动领域成功的可能。

Android的发布，使得移动设备领域的战争更加复杂。在这个领域，Microsoft本是先行者，2000.4.19号发布的Pocket PC2000是有据可查的最早版本。近10年的耕耘，微软在智能手机市场占有了14%的份额，而类同于Windows的开发方式以及越来越简单的.NET Compact Framework能让开发者迅速地迁移到移动平台。Palm是没落的王者，它一度是移动设备领域最具人气，也是拥有最多型号和用户经验的公司。而今竟把操作系统部分拆分成单独公司，Palm的手机上也跑起了Windows Mobile。Palm Pre的发布，不知道还能挽回多少开发者的心。Nokia可谓移动设备的巨头，占有大量市场份额，但是复杂的S60/S60 3rd/S90，以及令人眼花缭乱的UIQ的联盟，总让人觉得战略缺失、平台复杂。相比其他平台，Symbian的编程和复杂度，为开发者带来了不小的门槛。而对于最终用户来说，这种证书/签名更让人折腾不已，很难相信普通的手机用户能够成功地安装上软件。移动开发领域的新贵Apple，其封闭的开发平台，完全小众的Object C为开发者更是带来不小的难度，如果不是在Intel芯片的机器上可以安装Macintosh，怕国内能够开发iPhone的程序员会更少。

而Android颇为体贴地采用了Java。或者我们将其称为“类Java”的语言会更合适，据说他们完全重写了Java的虚拟机，使得执行效率相对较高。而开放的文档，完整的示例以及方便的调试平台，使得开发者能够迅速上手。对于最终用户来说，通过Android Market，安装程序异常简单，几乎是所见即所得。

在移动设备领域，需要的不仅仅是强势的操作系统平台，各种各样的小型应用才是真正推动和鼓励用户更加喜欢这个平台的原因。

为了撬动市场，Google做了一个令世人吃惊的比赛。2007年11月13日，

悬赏1000万美元，邀请开发者为业界第一个完全开放并免费的Android平台开发移动应用。这个举动，更是另外一个关键的引爆点，迅速地把开发者的心拉到了Google的平台之上。和苹果遮遮掩掩然后一鸣惊人的动作完全不同，Google在样机还没有出来，仅有SDK和模拟器的时候，就先声夺人地把开发者吸引了过来。

2008年10月，Google的第一台手机G1终于发布了。一时间风头无量，不仅仅是HTC, 三星、摩托罗拉、LG将合作开发其他型号的Android手机。而中国移动，全球最大的移动运营商，也全面采用改版自Android的OMS作为其定制手机的核心操作系统。不仅如此，一直火热不已的上网本，也有即将跑上Android系统的传闻。

Android上程序的开发环境准备

对于开发者来说，操作系统再好，市场前景再广阔，最重要的还是在于是否能够快速上手开发。笔者依然清晰地记得十年前为了学习Java理解path、classpath等概念的过程，在不久前的一次聚会中，竟然还有人在继续抱怨入门的困难。

那么，Android上如何上手开发呢？只要如下几步：

1. 下载Android SDK.（http://developer.android.com/sdk/1.1_r1/index.html）目前最新的版本是2009.1月发布的1.1 r1；

2. 在Windows/Mac/Linux下，解压缩到一个目录即可；

3. 安装Eclipse，安装一个叫ADT的插件即可；

4. 在Eclipse上即会出现Android的相关项目创建向导；

5. 在编写完Hello world之后，可以直接选择Eclipse插件上预置好的Debug来启动模拟器进行程序调试（图1）；

6. 如需使用其他IDE进行开发可参考http://developer.android.com/guide/developing/other-ide.html。

如果对Eclipse比较熟悉，理论上配置起来和试运行第一个程序不会超过5分钟。当然，除了会写Hello World，还需要真正了解Android的相关开发，我们还需要一些附注的例子来对相关的操作方式进行学习。

同样，Google也提供了一些相关的示例代码供开发者参考。在安装完SDK之后，可以在<sdk>/sample/目录找到关于API的Demo, 一个记事本的程序来帮助开发者理解Android中特有的一些概念。

Android淘金之路

1. 创意的产生。创新、创意几乎会让每个有志于自己做点东西的开发者感到头疼。到底应该做点什么？做什么用户才会真正喜欢？

用“淘金”来对比Android上的创意过程似乎会比较合适。最古老的淘金过程就是挖出一桶桶原始的金土，再把金土倒到过滤槽中用水冲洗，当沙土被冲走之后，剩下的就是金子和比重较大的物质，最后，用特制的洗砂工具过滤到其他物质之后，才能剩下一点点的金砂。

而在Android上做点应用也是如此。可能会看到成千上万的可能，也会突然兴起成千上万的想法。但是，真正筛选之后，剩下的可能什么都没有。也许会出现一颗令人兴奋硕大的金砂。

在笔者周围，也有这样的一群朋友，他们也在绞尽脑汁地构想着相应的想法。而实验出的结果和思考的方式大相径庭。其中一位是IM-Easy的作者，这款产品荣登Google千万大赛的前50名。而他的想法的起点只是单纯地想做一个更加好用和花哨的IM工具，没想到现在的下载量已达到近10万。另一个朋友是最近网络上沸沸扬扬的iPhone分歧终端机的作者。而这个想法的产生，却仅仅来自于一次冲动，纯属好玩。没想到传上去的第二天竟然有编辑专门为其做了教学视频。弄得做手模的小姑娘兴奋不已，一直后悔没有在手上画个专有的logo。

图1

图2

图3

图4

而笔者开发的“掌词”，一个Android上在线查词程序想法的诞生，来自于对一款名叫Free Dictionary的词典程序竟能在免费的下载量上排名第一的费解，而且功能想法的诞生源于如

下几个方面：

① 各大Market的排名。从排名上，就可以看到最受欢迎的程序究竟长什么样。没准就能从中挖到好的灵感，然后构建自己的产品。

② 他山之石。智能设备发展了这么许多年，历史上已经积累了种种好用和实用的软件，也有不少人总结了“xx平台上必装的xx种软件”。在新的平台上没有，也许就是一个新的机会。

③ 神仙会。所谓“三个臭皮匠，顶个诸葛亮”。也许是人家中无意中的一句话，却成为了你的一个想法。笔者的一个朋友写了篇“一坨狗屎引发的创业”的博客，也许就是你不经意中地多想了那么一下，在聊天中就诞生出了一个好的创意。

2. 产品的设计。一次朋友聚会，言谈中提及现在业界正在兴起“学习化腾好榜样”的风潮。旁人大惊，原来故事来自于网上流传的“马化腾关于产品设计与用户体验培训”的一篇帖子。其中提及了三个核心的概念：产品设计、产品运营和交互设计。

作为一个开发者，可能很少会去考虑到作为一个“产品”到底应该长什么样？要么就是客户说了算，要么就是老板说了算，换做自己说了算，这不免会让人相当纠结。

但这又是作为一个独立的矿工所必须面对的问题。在矿山选定之后，自己来觉得要筛多少次，自己来决定要用水冲多少次。没有人会给定一个具体的标准，也许筛的多，冲的多，得到的会多。也许筛的多，反倒好东西都筛走了。

在“掌词”这个小程序的设计过程中，笔者也经历了这个过程：

① 产品的操作方式定位。在确定了做词典之后，输入、智能单词补全等这些功能都好定义，毕竟是大家已经习以为常的功能。但是，如何输入和如何显示结果却成为了一个大问题。试想如下的场景：你正在用Android的Mail程序或浏览器阅读文章，出现了一个不认识的单词，你将如何操作？本能地，我们希望启动词典程序。一般来讲，在GPhone手机上要启动一个程序，得经历如下过程：按home键退出当前程序，回到主屏幕点击一次；在主桌面上或者可以滑动的左右桌面上，找到相应的图标（图2）。或者通过下面拖起的菜单，找到图标（图3），最少点击/动一次，一般情况平均滑动和点击三次;等待程序启动，然后键入单词。如果是GPhone的熟手，可能会偶然地发现到，长按home键，会弹出一个小窗口，上面可以选择最近启动的一些程序图标（图4），这样，仍然需要两次。

在程序启动之后，原来的程序界面就消失了。往往我们会发现，我们是因为不认识这个词才做的查询，但是，现在点击了几次，页面切换之后，我还能记得这个词吗？不错，这样子我们就发现，这样的操作方式之下，如果来做词典查询，基本上不可行。沿着这个思路，笔者在开发的过程中，做了如下的考虑：如何让用户启动程序更快？最少，让用户的大脑还能缓存住这个单词的模样。不会在多次的复杂操作过程中迷失方向。在这个过程中，问题发生了转化，可能真正的问题是，如何让用户根本就不用记得这个单词的模样？

想到这一节，问题就变成了，是否可以设计出个透明窗口而不离开当前程序？另外，如果实在不成，强行将当前屏幕拷贝下来，作为查词词典的背景。这样，用户总不至于记不住了吧。

最后，在探索的过程中，确定了两种方式：从Google的提示条上启动程序（图5）；从home键长启动程序，在程序启动之后（图6），显示的仅仅就是一个非常小的查询框供用户输入单词。另外，用户触摸任何地方，当前程序都会退出（图7）（图8）。

这样，这个程序就设计成为了一个插件，不会干扰到用户的当前操作，又能实现用户的需求。

② 程序的核心架构设计。在Android上程序的设计和传统的桌面程序开发略为不同在于，功能和操作方式的确定决定了程序架构的设计。Android设计为一个单进程的操作系统。用户无法自主控制进程的生命周期，由Android系统自动接管，当Android系统觉得内存不够或者需要空间的时候，就会自动选择一些相应进程

图5

图6

图7

图8

进行中止。因此，在程序的设计中就会涉及如下几个问题：是否需要保存上次历史数据；如何设计程序可以让程序长驻内存从不中止。在Android上，存在类似Services的概念，然后通过Notification Bar的方式来进行消息通知。比如说，从Android Market上下载程序的程序下载通知，就是采用如下方式实现。因此，为了保证“掌词”的启动速度，在架构上采用了services + notification通知的方式来进行实现。基于如此架构，“掌词”还实现了监控剪贴板的功能来变相实现自动取词的功能。

3. 产品发布与收益。丑媳妇终归是要见公婆的。自己关在家里偷着乐的想法，终究还是需要面对市场。最终用户会毫不犹豫地用脚进行投票。

① 产品发布。在将产品发布到Android Market上之前，开发者必须根据这里的指引http://market.android.com/publish/signup 在Android上注册一个账号，并且需要缴纳$25的费用，才能有资格进行发布。只要有可以支持外币的信用卡，支付这部分费用不会成为太大的障碍。程序安装包在上传之前还需要开发者对其进行数字签名。

Android Market的程序发布速度可以称得上是极速，一旦apk安装包通过了网站上传空间的验证，就可立即通过手机访问Android Market访问到开发者刚上传的程序。

同时，一个专业的第三方Android Market程序介绍网站（http://www.cyrket.com）也会实时的收录开发者上传的软件介绍和下载地址，并声称提供二维码方便其他用户下载。

② 产品收益。也许大家最关心的就是：做这个产品将会得到如何的收益。目前来讲，有如下的两种收益方式：

● 为你的程序确定一个价格，然后发布上去。等待用户为你的程序买单。Android Market在2009年3月底支持了付费的程序上传。发布收费程序之前还需要注册一个Google Checkout账户用于接收收入。定价策略方式，一般都是以.99的方式来确定你的价格以刺激消费者买单。开发者最终可获得70%软件销售的提成。

笔者知道的有几种常见的方式：

定价为$0.99。这个是最常见的定价方式，按照一个外国朋友的说法，在支付$0.99的时候，基本上无需考虑，顺手就买了。如果程序不好，也不至于为了$0.99退款。

定价为$1.99—$9.99之间。据第三方数据统计，一般评价价格在$3。这个部分的程序一般开发了不短时间，另外同样有一个免费版有着大量用户量作为支撑，从而定价。这个部分的定位比较尴尬一些，笔者就听说过Android Market刚刚开始收费的时候，一个非常流行的免费软件改为收费之后，几个星期之内只有不到8个的用户量，并且还有4个用户要求退款。

定价为$999.99或者更高。也许大家觉得不可思议，的确如此。在以前iPhone上就有一个叫I am Richer的程序开出了这样的价格。而这个程序却出奇的简单，仅有一张画面，上面是颗硕大无比的红宝石。据说还卖出了不少份copy。后来在Apple的强行干预下，这个软件被下架了。在Android上，同样也听说了类似的程序出现。不仅如此，据说在开发者把价格由$999.99调到$0.99时，还引起了以前付费者的极大愤慨。当然，这个纯属和百万方格一样的另类，博君一笑尔。

● 广告。自从Google创立了AdSense的方式来为按照点击的方式计算收入之后，大大小小的网站站长的生存条件就改善了许多。同样如此，在Android上开发，如果会担心因为定价的原因导致没有用户量，选择免费版加上广告的方式也是一条出路。

目前来说，移动平台上最流行的广告方式就是Admobile。在iPhone和Android平台上都提供了相应的接口，以便用户调用。只要在相应的网站上简单地申请个注册码，就会拿到一段适用于你程序的代码。将代码嵌入到你的程序中，从而展示给最终用户。

但是，广告的方式有广告的弱点。第一，用户可能根本不会点击你的广告；第二，Admobile有自己的一套算法，并不会无限量地为你提供新的广告。

以下有几个数据供大家参考：一个朋友的程序，下载量近10万，日活跃广告展示量近30万次，但是广告的收益每天只有不到六七十美元；一个朋友新放上去的程序，前3日下载量近千份，日广告量展示量近3万次，而总共的收益也会平均下降到每天不到二十美元。也就是说，通过广告可能未必能够赚到你想象中那么多的钱。

4. 产品的推广。在互联网时代，信息的传播方式变得更加快捷和廉价。甚至不少人都开始抱怨信息过载的问题，但一条真正有价值的信息出来，可能就突然间就传遍了整个互联网。在产品推广上，无外乎以下几种方式：贴论坛；写博客；优化搜索引擎关键字。有幸的话，让一个知名的博客推荐下你。

当然，有钱人也有有钱人的玩法。如果有钱烧，也可以采用以下的方式：通过各大程序的客户端弹广告；门户站上打广告；在各大社区上精华置顶；在各QQ群拼命发帖；或者干脆预置到运营商的手机上。

展望

行文至此，笔者为数不多摸索经验的存货已经悉数倒光。笔者所开发的程序以及朋友所开发的程序，至今尚未创造出任何商业上的奇迹，也还在不断地接受用户的检验和认可。同时，也在绞尽脑汁地寻找着下一个更加有意思的想法和可能。无论是出于商业目的还是仅仅为了渲染个人英雄，各式各样的媒体仍然为我们书写着各式各样的财富传奇。

最终，仍然回归到一点：你的程序真的为用户带来了价值吗？也许，这才是最终金矿的入口。

移动应用在 Android 平台上的部署

■ 文 / 武海峰

2009年1月7日，重组后的中国移动、中国联通和中国电信分别拿到了工信部发放的3G移动网牌照。3G牌照的发放，突破了当前Internet移动应用发展中的网络带宽瓶颈。从协议的理论值上看，GPRS能提供最高至60Kbit/s的数据传输速率，EDGE的为236.8Kbit/s，而WCDMA中采用的HSDPA则能提供最高达14.4Mbit/s的数据传输率。

3G时代的到来，对正紧锣密鼓进行开发和部署Android的软、硬件厂商们来说，意味着更加坚实的市场。

Android平台的原型基于ARM处理器，并要求采用的ARM Core至少支持ARMV5TE指令集。而ARM926EJ-S是目前ARM公司提供的支持ARMV5TE的最低处理核心。换句话说，Android设计时，是以ARM926EJ-S及以上处理器的运算性能作为参考的。ARM926EJ-S的运行频率在200MHz左右。而更高的ARM核应用处理器，如ARM1136及ARM1176，最高的运行频率可达到约600MHz；ARM Cortex-A8的运行频率可达到600MHz以上甚至1GHz；TI最近宣布的OMAP4产品计划将采用ARM Cortex-A9 MPCore，设计目标是1GHz以上的运行频率。这样的运算性能，已足够满足手持设备的需要了。

当移动设备突破了运算能力和网络带宽的瓶颈后，其上的应用软件部署就成为了人们关注的核心问题之一。

移动应用软件部署的概念

移动应用软件提供商为了获得更广阔的消费市场，通常希望其产品能够运行在不同的硬件平台或操作系统平台上，并能兼容相应操作系统的多个版本，这样就使软件部署过程异常复杂。

移动应用软件部署，是使软件在用户系统上变为可用的过程中所涉及到的所有活动。从提供商的角度看，软件部署包含其产品在移动设备上可用到从移动设备上移除的全过程。

部署的过程

移动应用软件部署由如下环节构成：

● 发布：作为开发过程和部署过程的接口，通常由移动应用软件提供商完成；涵盖了将产品批量生产直至传送到用户端的所有准备工作，并收集和细化了完成部署过程其他环节所需要的信息。

●安装：涵盖了移动应用软件组件从软件提供商端传送到用户端的全过程（包括配置），为激活软件做好准备。

●激活：在用户端运行已安装的移动应用软件；对复杂的系统来说，可能还需要初始化其他服务和软件。

●反激活：激活过程的逆过程，关闭运行中的软件。在其他一些环节进行之前也需要执行反激活，比如更新。

●更新：作为一种特殊的安装过程，意味着新组件版本的部分直至全部的传送过程。在安装之前，大部分应用会要求关闭运行中的该软件。当然，部分应用允许在运行时完成更新。

● 适应：和更新类似，过程包括对之前安装的软件系统进行的修改；与之不同的是，更新通常是由远程事件触发的，比如软件提供商发布了新的组件版本，而适应则通常由本地事件触发，例如用户端软硬件环境的改变。举个例子，用户在使用移动设备时将网络接入方式从HSDPA切换到了Wi-Fi，这时系统就要通过适应来调节上层应用获取网络数据的来源。

● 移除组件：移除过程中必须监测用户端系统的状态，不能影响其他已安装程序的运行。一般可通过依赖关系检查来避免移除与其他应用共用的组件。

● 退役：软件提供商不再提供对该软件的支持。通常由软件提供商来向所有已知的用户发布这个消息，但不会直接影响用户继续使用该软件。

一些重要问题

在移动应用软件部署过程中，有些问题在制定软件部署的策略时必须考虑：

●组件依赖：软件系统一般是由多个相互之间存有依赖关系的组件组成的。依赖关系相当于组件之间生成的有向路径，上层的应用依赖于底层组件。所以部署过程要求必须能够正确解析和处理这种依赖关系，尤其是在安装和移除的过程中。

●内容传送：将软件包从提供商那里传送到用户的过程叫做内容传送。在带宽较低的环境下，通常要采用合适的方法和策略来保证有效地传送，以降低带宽占用。

●通用性：目前的移动设备基于多样硬件平台，运行着多种操作系统。部署过程必须考虑这些平台之间甚至软件版本之间的差异。

●部署过程的监测：通常部署过程都需要获得用户端的特权后才能进行，也就意味着可以访问系统中的任何资源。所以部署系统必须要提供在部署过程中执行了哪些操作，改变了哪些组件之类的信息。这些信息通常是通过日志文件和提醒服务实现的。

●无线连接：传统计算设备的软件部署通常通过有线连接来接入Internet，但移动设备一般不具备这样的条件，通常通过无线网络连接来进行软件部署，这个过程就称为OTA。常见的无线连接有Wi-Fi、GPRS、HSDPA、BlueTooth等，它是移动设备平台上软件部署的重要特征。

在Android上的部署

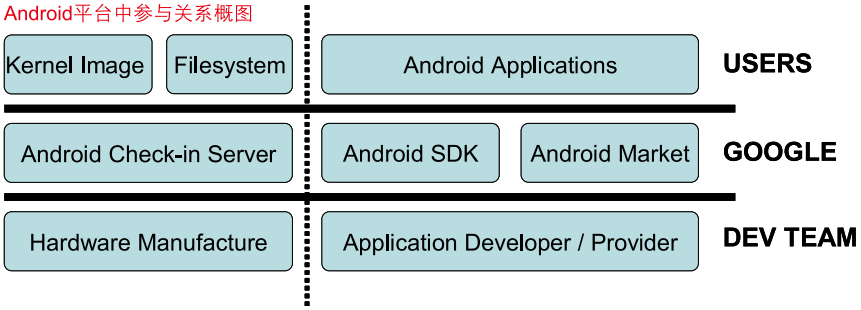
Android平台的软件部署综合了上述所有适用于移动设备平台的特征。Android的应用软件包是以apk文件（Android Application Package）的格式分发传送的。虽然每个apk文件都包含了应用程序的字节码（.dex文件，即Dalvik Executable Files）、资源(Resources)、固有文件(Assets)以及清单文件（Manifest File），但由于Android架构中的中间件对底层库进行了完好的封装，软件应用对底层的软件包就没有了直接的依赖关系，所以在更新时仍以完整的apk文件为基本的传送单位，这和Windows上的应用安装类似。

在提交应用之前，应用软件开发人员或提供商需要注册成为Android Market开发人员。当应用软件开发人员完成开发测试，准备发布应用程序时，首先关闭调试用途的各种日志，清理数据并标定版本，然后编译，SDK将自动把相应的Dalvik字节码文件等打包为apk文件，最后用私钥签名，提交并发布到Android Market。当用户感兴趣并确认需要安装后，付费并从Market下载应用程序，最后安装到移动设备上。由于这个过程中没有审核的环节（AppStore应用在提交时包含这个环节），所以被认为是Android应用安全性的潜在威胁之一。

底层系统软件一般由硬件生产厂商或底层软件厂商开发维护，在有更新后将通过Google发布在Check-in Server上，供终端自动更新系统时下载使用。在更新底层的软件组件，特别是Linux Kernel和文件系统时，一般以映像文件格式更新。

目前Android在部署系统软件时还没有统一的策略，依赖于各个厂商的实现，比较一致的是，不管是应用软件还是系统软件，在部署时都需要软件仓库来存储针对各个设备平台硬件特性的软件版本。应用软件集中在Android Market存储和管理，而系统软件则集中存储在Check-in Server上。Google提供了一个专门的指向Check-in Server的HTTPS URL，终端设备可通过这个链接，安全地与Server之间传递信息和软件包。终端和Server之间通过JSON来描述硬件平台特征信息以及软件版本、依赖性、资源等信息。

Android平台中参与关系概图



上图说明了各层软件提供商及用户与Google之间的关系。Google处在中间的位置，衔接了上下游开发团队以及用户：一方面负责维护硬件厂商存储系统软件的Check-in Server，当硬件厂商付费后就可把带有硬件设备描述信息的系统软件映像文件上传；一方面向应用软件开发人员提供SDK，维护存储Android应用的Android Market，并负责Market应用软件开发人员的注册、收费以及应用售出后的收入分配；一方面和用户交互，通过Android Market向用户提供应用并根据其是否收费来决定用户是否支付费用。

可以看出，这样的部署框架里，Google处于最核心的位置：它主导着产品开发过程中的整体架构，全权控制着SDK的版权和内容；占据着整个部署过程中的轴心，分享着用户支付给上下游厂商的利益。用户处在最上层，他需要做的就是付费或免费从Check-in Server更新系统软件或者从Android Market下载应用程序；而开发人员处在第一线，一部分（硬件厂商或手机厂商）负责维护可用的系统软件（操作系统和中间件），一部分（应用软件提供商/开发人员）开发应用程序。

当然，系统映像文件和应用程序未必一定要存储到Google的Check-in Server和Market，有实力的厂商也可以选择构建自己的Server，或者如果第三方愿意提供类似的服务也是一种途径，只是如果没有一个整合统一的更新过程，构建这样的服务可谓困难重重，更何况整个软件架构都由Google掌控，几乎对其中任何组件的修改都要有Google的同意才会被集成进Android。

写在最后

分析了Google在Android在移动设备的部署中所占有的独特位置，我们不难看出，软件部署就像软件产品的生命线，一头牵着硬件设备厂商，一头牵着软件提供商，一头牵着客户；在Android平台的部署实践中，更是协调了整个产业链中的各个环节。

Android开放的架构，精心为移动应用定制的类库，强有力的社区支持以及Google在Internet应用上的丰富经验必然会推动它在移动设备上的部署。■

作者简介

武海峰，专注于移动计算平台的启动引导和系统集成；毕业于西安交通大学信息与通信工程系，现在风河系统中国研发中心工作。

智能手机应用安全现状及前瞻

■ 文 / 王颖奇

智能手机越来越被大众所接受，而用户根据以往使用电脑的经验，往往认为智能手机也应该和电脑一样，有病毒、恶意软件，木马等等。而对研发人员来讲，也理所当然地认为有智能手机软件安全这个领域，然而这个领域和电脑上的情况是否类似？用户是否需要一个手机安全软件？手机安全软件的市场方向如何？或者如何使开发出来的手机应用更为安全呢？在回答这些问题之前，我们可以先看一下目前市场上主流智能手机操作系统的安全特性。目前主流的智能手机操作系统有Symbian、Windows Mobile、RIM、Palm、iPhone，Linux等，这里仅就其中几个作以介绍。

几大手机操作系统的安全特性

先说iPhone，一个iPhone如果想要安装软件，只能到苹果官方的软件商店上搜索。而苹果对软件的功能和安全性的审查极其严格，虽然有人对这种封闭、垄断行为很是不满，但是对最广大的普通用户来说，至少在安全这个问题上，iPhone的用户根本不用操心。

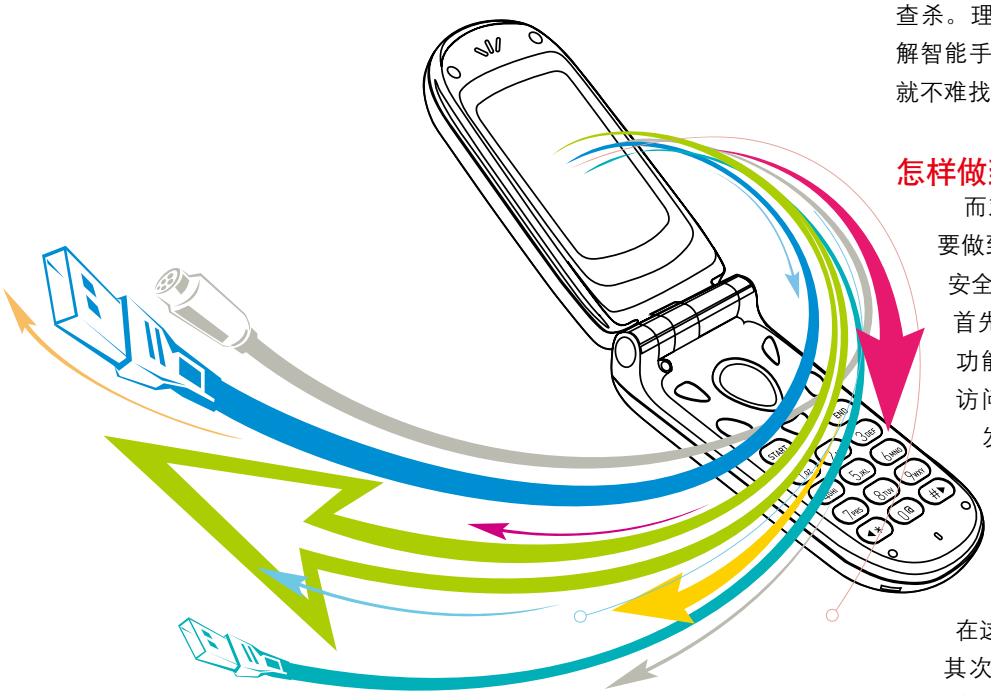
Android，也就是Google推出的智能手机操作系统，目前也是沿用这个策略，用户只能安装官方软件商店上的软件。而Symbian是一个比较复杂的系统，市面上能见到的有Symbian S40，S60第一、第二、第三、第五版，UIQ等。Symbian S60第三版以及之后的所有版本，包括UIQ，在上面正常安装的所有软件都必须通过Symbian官方进行安全认证。Symbian虽然没有官方的软件商店限制，但是官方安全认证这一点就已经给S60高端智能机一个很好的安全保证。不用说做一个病毒传播出去，就连做一个正常的软件想要发布都会有很多门槛：必须购买一个200美元一年的开发者资格；每次发行一个版本都要付给官方20美元认证后才能大范围地安装使用。要想写一个恶意软件很容易，但是要装到很多手机上来用并传播开来，基本上不可能。至于S60的第一、第二版本，诺基亚在2005年的N90以后就再也没出过这两个版本的智能手机了。他们的安全认证等级比较低，市面上已经没有产品卖了。

在智能手机领域，大家可以认为Palm OS已经溃败退出，近来要推出的Palm pre会搭载全新的Web OS。目前没有看到真机，但相信这种现代的操作系统，会十分注重安全性。至于Linux，其实手机上用的很少，摩托罗拉一直在使用，单从血统来看，Linux的手机安全性肯定不必担心。另外，像这种比较偏门的机型，目前占有率不高，未来发展方向也不很明朗。而这一点反而使得Linux操作系统的智能手机更加安全。Blackberry，也就是RIM的软件安装，目前未见到严格的安全认证限制，同上面未说完的Symbian S40一样，是目前在市场上占有率较高且有中毒风险的系统，而Windows Mobile的安全认证机制基本是空谈，也属于高危范围。

智能手机安全现状

这样看来某些智能手机还是有一定的危险性，那为什么目前没有很多手机病毒呢？其实，目前在电脑上，单纯的病毒已经无利可图了，病毒已经不再是一种技术的发泄和炫耀工具，病毒作者已是无利不起早的经济利益偷窃者，电脑上的网游盗号，网银盗号才是他们要关心的，手机上没有他们想要的，我们认为重要的亲密短信，隐私图片，对犯罪者是没有吸引力的。这也是目前在这些相对危险的操作系统上也没有手机病毒爆发的重要原因。

说到这，可以看一下现有智能手机安全软件情况，目前针对智能手机的安全软件大概分两种，一种是杀毒类的硬安全；另一种是隐私保护或恶意电话屏蔽等的软安全，如果说还能找出一种的话，就是前面两种的结合体。手机杀毒软件实际上目前处在雷声大、雨点小的状态，手机操作系统有一个很重要的概念就是任何软件都只能在特定操作系统平台上运行，也就是说Symbian的软件不可能在Windows Mobile的手机上运行，iPhone的软件也不可能在Android上跑。实际上一些Java的程序是有可能在各种不同的智能手机平台上传播的。但是有一个前提要说的是，Java程序能做的事情很有限，访问网络，访问



手机上的文件，发送短信这些都会有十分明显的提示，让用户确认后才会进行。这个安全限制是在Java这一层次就已经解决了。

因此，如果要做一个智能手机杀毒软件的话，只可能针对特定平台来开发，并且要有专门的人来负责该平台的病毒样本收集、分析、特征提取等。就前面讲到的各个智能手机操作系统的情况来看，实际上在国内目前只有Windows Mobile的杀毒软件才有可能有一定的用户需求，而从真正的手机病毒流行情况来看，这个用户需求也只停留在“有可能有”的阶段。现在已知的手机病毒不过300种左右，而且是分布在各个平台之上，甚至大多只能运行在老版本的操作系统平台上。由于目前大部分平台良好的安全性限制，手机病毒没有爆发的迹象。市面上能看到的手机杀毒软件的实用性非常有限，甚至有些厂商有故意控制舆论导向，鼓吹手机病毒爆发的嫌疑。作为一个有良知的厂商，不应该从该方面牟取用户钱财。

用户的真实需求

那么智能手机安全软件是否真的无事可做呢？手机作为个人通信工具，每天跟随用户，保存了很多个人隐私。并且近年来收费电话，垃圾短信，电话和短信诈骗等层出不穷。如果能从这些方面着眼，解决用户的实际问题，才是目前真正要做的。而这一类的软安全软件目前市场上也有很多，比如，一些来电显示软件，就可以很准确的告知用户来电的归属地，一定程度上杜绝了电话欺诈；另外一些软件可以由用户自定义号码黑名单，主动的屏蔽掉骚扰电话和短信。

当然，也不能保证将来的智能手机操作系统就不会有杀毒软件的真正需求，这个要看各大厂商对安全的重视程度。理论上讲，哪个操作系统厂商不重视安全性，哪个厂商的平台就会有被淘汰的风险。软件病毒爆发和流行的根源在于操作系统平台厂商的设计失误，第三方软件永远只是亡羊补牢，不具有杜绝病毒的能力。

此外，智能手机软件绝不是将电脑上的软件移植到手机上，而是要针对手机的特性做全新的设计和产品规划，因此，智能手机安全软件绝对不是学电脑安全软件的文件监控和病毒查杀。理解手机用户的真正需求，了解智能手机操作系统平台的安全特性，就不难找到这个领域的正确方向。

怎样做到安全

而对于一般的智能手机应用软件要做到安全，或是说被用户认为是安全的，大概要从几个方面着手：首先，在任何会造成用户付费的功能上给出明显的提示，比如：访问网络造成的数据流量费用，发送短信造成的信息费用，软件本身的注册产生的费用都该及时明确地给用户提醒，这是基本的商业道德，国内的几个流行的智能手机产品在这方面都作了很好的示范。

其次，作为一个商业软件，要打数字签名，从被授权的证书厂商处购买数字签名证书，一般的费用是一年几百美元，在初次购买的时候会有很严格的开发商身份验证机制。而这个步骤从市场推广和用户体验来讲是必需的。

最后，利用平台固有的销售和推广渠道是一个双赢的事情，比如现有较成熟的iPhone的AppStore、Android的Market，以及会相继推出或完善的Blackberry的AppWorld、Palm的App Store、Windows Mobile的Marketplace、Nokia的Ovi等，传统的找软件方式将会被AppStore模式取代，这种目录式的软件查找方式填平了软件开发者与用户之间的鸿沟，为用户方便地找到合适的软件提供便利，而更重要的是，这些软件在用户的心目中是安全的。

作者简介

王颖奇，2004年加入金山软件，参与WPS V6研发，曾任金山毒霸水银平台项目经理，关注智能手机研发。

AppStore 模式下的移动产品推广销售策略

■ 文 / 项有建

记得《程序员》早在2008年10月刊的杂志上，就已经提出了AppStore模式的概念。如今，这一模式已经得到了业内的普遍认可，并为诸多商家带来了巨大的盈利。在此，笔者将从人们所熟悉的传统产品的推广和销售方式说起，深入地分析AppStore模式下的移动产品推广、销售策略，希望能对广大移动开发者有所启发。

数字产品推广销售传统模式

随着互联网的出现，伴随而来的是电子商务的产生，随着技术手段的变化，使得产品推广和销售模式发生了重大的变化。首先表现在长尾理论的出现，推翻了传统的28理论。

而Google则是长尾理论的最佳实践者：Google有效地利用了长尾策略，通过Adwords广告使得无数中小企业都能自如投放网络广告，而Adsense广告又使得大批中小网站都能自动获得广告商投放广告。利用Adwords和Adsense汇聚成千上万的中小企业和中小网站，其产生的巨大价值和市场能量足以抗衡传统网络广告市场。

数字产品可以方便地通过网络进行实时地几乎是无成本地传输，这个特点，使得数字产品在销售方式取得了实质上的突破，也就是说，它具备了长尾理论生效的前提条件。

手机应用产品推广销售的传统模式

在AppStore模式出现之前，让一款手机软件流行的最佳途径是说服手机厂商在手机出厂之前就预装这种软件，这是一种非常艰难的工作，成功者并不多。之前，手机的用户们很少自己下载程序到手机上，造成这种现象主要有两个原因；一个是原来的手机大多采用的是封闭式的平台，根本不允许用户自己安装所需要的软件；另一个原因就是，可以上网的手机数量并不多，并且费用高、网速慢。

笔者在2004年，曾经创办过一家公司，产品是活码手机输入法。产品的品质在当时还算不错，但由于公司实力小，地处偏远的广西，无法成功地说服手机厂商对活码手机输入法进行预装，所以，几年来连一份产品也没能卖出，活码公司已于2008年基本倒闭，但活码手机输入法的电脑版（免费）已经在不知不觉中有了二十多万的用户。能够让众多的电脑用户用手机输入法的实例，说明了产品品质的优良，但整个事件说明的却是说服手机厂商在手机出厂之前就预装第三方开发软件的难度之大。

AppStore模式的出现

苹果的AppStore——“iPhone的软件应用商店”为第三方软件的提供者提供了方便而高效的一个软件销售平台，他降低了人们进入手机软件这个领域的门槛，使得第三方软件的提供者参与其中的积极性空前高涨，适应了手机用户们对个性化软件的需求，从而使得手机软件业开始进入了一个高速、良性发展的轨道。苹果公司把AppStore这样的一个商业行为升华到了一个可以让人效仿的经营模式，可以说是苹果公司的AppStore开创了手机软件业发展的新篇章，AppStore无疑将会成为世界手机软件行业发展史上的一个重要里程碑，其意义已远远超越了“iPhone的软件应用商店”本身。而Google将网络的概念——“开放与服务”引入了AppStore，使AppStore升华成为

AppStore模式。

纵观AppStore大战中的玩家，可以这样总结：以操作系统作为支撑平台的有苹果、Google、诺基亚、微软和黑莓，而主要作用于开发者运营商的类型目前只有中国移动一个，并无真正属于自己的操作系统——中国移动拥有接近无限的客源，缺的只是一个向客源实施辐射的平台；具备了网络平台的有，Google的搜索引擎、微软的浏览器门户网站、传统下载站。以原有的网站作为平台，一般手机生产商型由于无法找到对应用AppStore进行有效支撑的平台，所以其威力较弱。

我们不难得出结论，拥有主流操作系统、拥有全方位网络辐射能力（Google的搜索引擎、微软的浏览器），拥有近于无限用户资源的主流运营商，将在AppStore之战中占据优势地位。也就是说，如果不出意外，AppStore的王者，将在中国移动、诺基亚、Google和微软中产生，而苹果则由它的粉丝文化支撑起自己的一片小天地。

最佳的推广销售模式

AppStore模式的出现，改变了原有的手机软件销售模式——从厂商预装转变为自由销售。减低手机厂商的软件成本——在手机出厂时每增加一款预装软件都意味着软件成本相应增加。AppStore模式的出现，使得手机也将像个人电脑一样，在出厂时只装上基础软件，其他个性化的软件由用户自己安装、购买。

由于软件成本等原因，手机厂商不可能预装过多的软件，由于手机厂商对第三方软件的垄断性，极大地限制了整个手机行业的发展，AppStore模式的出现，降低软件制作商的门槛和成本，使得软件制作商可以面向最终用户，之前他们只能面向中间商，使得整个手机软件行业的发展受到了极大的限制。

随着3G在全球范围内的应用和普及，在行将到来的手机网络时代里，人们可以通过称为手机的数字终端享受原来只有在电脑上才能享受的网络服务，随着人们对网络的认识逐渐加深，网络能够给人们提供的服务越来越多，网络给人们带来的帮助也越来越大。相关的数据表明，智能手机的销售今年呈强劲的上升势头，今年第二季度它占据全球手机市场的19%，比去年同期增长了9%。随着智能手机的日益普及，智能手机操作系统也越来越受到重视。随着2G手机的3G化，人们对手机个性化的追求使手机软件产生了大量的需求，而具有3G概念的手机又给这种需求成为了可能实现的现实，随之产生出巨大的一个市场，而AppStore模式正是一个为这个市场而生的优秀市场模式，也就是说：需求催生了“软件销售和传播”市场，AppStore则是为这个需求找到了一个适合的模式。

可以预见，AppStore模式将对整个IT行业的发展趋势产生重大的影响，其影响主要表现在几个方面：一个是将改革原有的手机软件销售模式——从厂商预装转变为自由销售，对手机厂商来说，将更加专业化，作为厂商在手机出厂时只需安装基础软件，而个性化的软件由用户自行选用，使得手机个人电脑化。对于AppStore平台拥有者来说，他得到了一个渠道，一个行业的制高点，软件制作者来说，得到了自由发挥的一个舞台，对用户来说，以更低的成本得到了更为人性化的服务。

第三方开发者的赚钱秘诀

以苹果的AppStore为例，开发者只需要好好的做自己的app，销售、收款、物流、交易和发布渠道全部交由Apple搞定，分成的比例为营销收入按三七分账：开发者70%，Apple占30%。

首先，开发者要购买一年99美元的开发者license，他的入口是苹果iPhone Dev Center上的一个链接。license审批通过之后，开发者就可以正式进行开发工作了。当开发完成后，要将其提交给Apple审核。提交步骤十分烦琐，要有很多不同尺寸的截图，还要有一定的格式。N多表格填好后，接下来就是等待审核的过程了。一般在5天左右会有回复，Apple不满意的地方开发者要顺着他们的意思进行修改。此外，Apple还有一个合同审核的步骤，不管是销售app还是免费的app，都要通过他们审核提交者的纳税合法身份，这是一个人工的步骤。

值得一提的还有，苹果公司拥有一个“取消开关”，可以跳过手机上不需要的程序。每个iPhone都包含一套代码，理论上可以从手机上任意删除软件。苹果公司称这对于解除恶意程序是必要的：尽管“我们从来无须推动那个控制杆，但如果没有那样一个控制杆，出了问题我们将不负责任”。到目前为止，苹果公司已经取消了其商店中的几个应用程序。

像Sun工程师Ethan Nicholas辞职开发AppStore应用，而最终成为百万富翁的例子毕竟还是少数，更多的app产品还是无人问津。由于AppStore商店的特殊性，开发者在将自己的产品放入AppStore的货架上后，就对它失去了控制。所以，如果开发者希望推广自己的产品，只能“曲线救国”，更多地考虑在AppStore之外的相应推广策略。

作者简介

项有建，主要从事人工智能、搜索引擎、3G相关领域研究，3G理论体系研究，专利作品：活码手机输入法、梦幻ME手机浏览器。

将移动市场细分

■ 记者 / 圣伟

随着苹果的移动互联网终端软件商店AppStore获得巨大盈利之后，Google、微软、NOKIA、三星也都在纷纷效仿其运营模式，中国移动也即将推出Mobile-Market。那么，这种新兴的移动应用销售模式，究竟会给整个移动生态链带来怎样的影响呢？又会给创业者带来哪些机会呢？本刊记者近日采访了一位这个领域之中的创业者——北京空宇数联信息技术有限公司总经理孟闯先生。

孟闯的创业项目——SkyGame游戏商店，是一款基于MID/NetBook的游戏平台，用户可以利用其管理MID/NetBook上的游戏，包括安装、升级、下载等。另外它也是一个游戏发布平台，软件开发商或者是个人开发爱好者，均可以通过管理平台免费提交游戏资源，用户可以通过搜索及浏览的方式下载游戏。SkyGame游戏商店中提交的游戏均经过严格地测试和多重加密处理，确保了其实用性和安全性，同时智能匹配用户所使用的超移动电脑类型型号，只显示用户可用游戏，避免了繁杂的寻找与自己机器匹配游戏的过程。

软件商店，服务商店

孟闯认为，软件商店是一个笼统的概念，称其为服务商店或资源商店更为合适，因为其中不仅提供各种应用软件，其实也在提供一些特色服务，比如天预报、汇率计算、出行指南等。虽然AppStore的运营模式备受关注，但因为各种软件、各种服务、各种资源数以万计，如何协调、测试、运营、管理、呈现这么多资源，不仅仅是一个交易平台技术开发的问题，更是一种服务的提供。所以真正能够有资格和能力运营软件商店的运营商很少。

随着AppStore模式受关注度的逐步提升，在2009年世界移动通信大会上，诺基亚宣布正式推出应用程序商店Ovi，几乎在同一时间，微软也宣称其推出的Windows Marketplace for Mobile将成为“一个集搜索、浏览和购买手机应用程序为一体的市场”，并表示程序开发者将可“毫无限制地提供应用程序”。

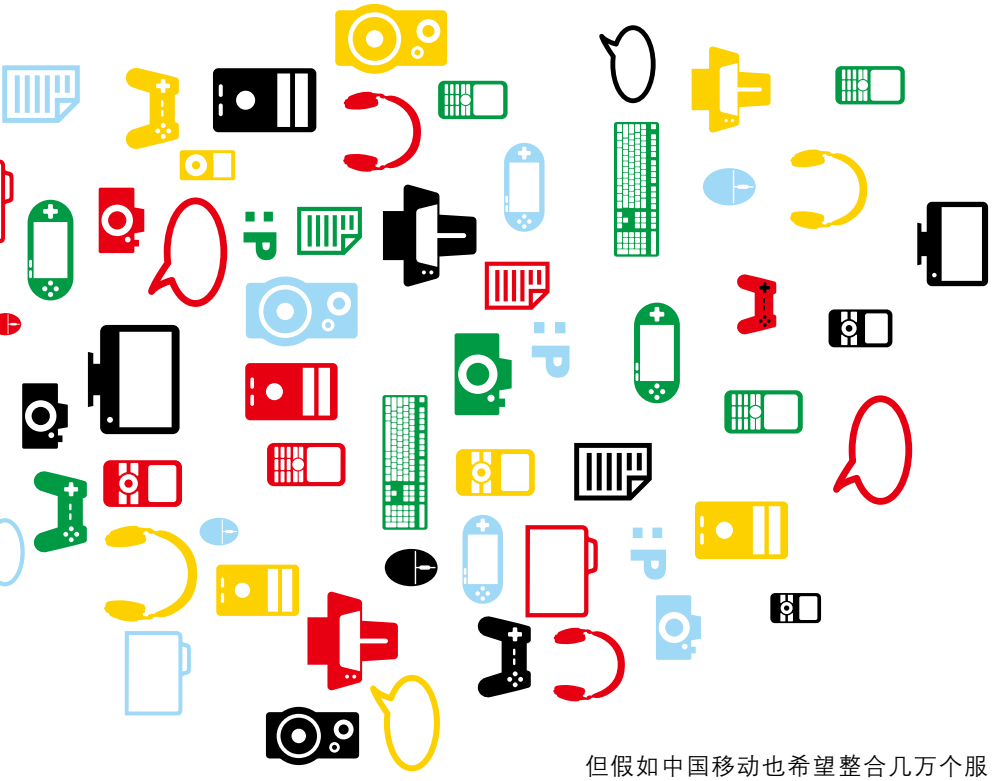
2009年1月29日，三星App-Store的试用版已经在英国开始运行了，并且计划在今年下半年推出全球化的版本。通过三星AppStore网站发布和推广应用是完全不收取任何初始费用的，关于收费应用的分成模型，目前采用的是以R/S（Revenue Share）为基础的分成模式。

中国移动的软件商店模式，应该也和AppStore大同小异，并且所有的运营行为都基于它的自有资源，包括定制的手机和上网本等。通过这些资源，就能通过控制渠道，进而控制客户流。经过一段时间的软件商店运营，他们会发现那些软件用户比较感兴趣，并会在主流的定制手机上整合这些软件，反过来促进这些自有资源的销售。

将市场细分

孟闯很早之前就有了创办这样一个软件商店的想法。甚至在苹果刚刚发布AppStore的时候，他就已经做出了相似的雏形。但当时他并没有对外大规模地发布。这里有三点原因：第一，既然称之为软件商店，那么上面就应该摆放着很多应用产品，但当时他开发出的产品并没有很多用户可用的软件及服务，只有MID上或NETBOOK上的十几个应用软件，所以这个商店去推就没有意义；第二，运营这件事不是一个刚开始创业的小公司能做的，他需要运营者掌控的资源或整合的资源比较多，比如足够的客户量，而这些对于苹果这样的公司就很简单，他们拥有自己的设备和操作系统，所以捆绑了大量客户资源，会形成良性循环；第三，在运营方面，如果一个小公司仅仅公布了一个自身制定的协议，没有多少开发者会愿意遵守一个不知名的协议标准。此外，随便谁来开发这个东西，都能上线卖是不现实的，需要有各种审核，各种流程的控制，及相应的功能验证和测试，甚至要判断内容上是否违反国家的政策，这些事都不是小型的创业公司可以做的。

所以，孟闯经过思考，决定将市场做以细分。他认为，第三方的小公司应该从一些细分的点切入，不要一开始就做一个“超级沃尔码超市”，那样很难运营起来，而建设一些专业的、



细分的小超级市场更能实现以小博大。虽然琳琅满目做不到，但某一个更小的市场领域，比如基于Moblin，创办一个MID休闲游戏商店，就可以作为一个很好的起步点。之所以选择Moblin，是因为MID这个概念最初就是由Intel提出的，而Intel的Moblin也完全是为了支撑MID这个市场做的，它是一个开源的、多方参与的，多方遵守的运行在MID设备之上的操作系统。并且，整个Moblin系统是MID上软件运行的一个基础，这对于我们的开发来讲是非常重要的。

孟闯的计划是，第一步就从休闲游戏商店开始做起；第二步向联网游戏商店进军；第三步，发展收费游戏，通过一个逐步演变的过程逐渐做大做强，最终实现他的软件商店梦想。

市场细分的机会

那么市场细分究竟会给刚刚创业的小公司带来哪些机会呢？

虽然中国移动还没有做出真正实质性的动作，但我们先假设其确实推出了自己的软件商店并进行了运营，而他的运营方式在某种程度上会与AppStore模式有所区别。基于iPhone的应用现在已经有几万个正在运行。但假如中国移动也希望整合几万个服务的话，将有大量的工作去做，而这些诸如每个软件的测试和内容审核等工作中国移动会自己做吗？孟闯认为，中国移动很可能也会采取类似于之前2G移动时代的合作模式，即专有的软件商店SP（服务提供商）会出现。比如，某个SP只提供游戏类的软件服务，而另一个SP只提供天气预报，还有提供办公软件的等等，这相当于分担了一部分中国移动的工作量，所以这其中就存在着细分所带来的机会。因为SP提供了整合、测试、运营、维护的工作，自然会从中分享收益。按以上分析来看，SP应先练好内功，做好相应的细分服务项整合开发工作，为真正的合作运营铺垫，创造细分的专业化市场机会。

软件商店的门槛

如果想要把软件商店做好，其实还有很多“门槛”和“运营思路”在里面。孟闯认为，至少具有以下几点特征，才可真正称之为一个理想的软件商店：

首先，它应是最终用户获取移动互联网资源最快捷、最智能的的方法。用户通过相应设备上软件商店所看到的资源，包括软件、游戏等，可以方便完成软件的安装、删除及更新，而不必担心软件的兼容性及依赖性，软件商店自动分析其操作系统及硬件规格，根据操作系统及硬件规格显示其可用的资源列表，并分析软件包的依赖性，对于缺失的软件包，会自动从服务器上下载并进行安装。

同时，要保证用户在成千上万个应用中找到自己所需要的应用。这就涉及要对每一个用户做个性化的通知和服务。比如，某一个用户在使用软件商店时可能只看新闻类的资讯，或只看股票行情，每个人都有不同的需求点。那么，作为软件商店的运营者，所要迈过的“门槛”就是做一种匹配的技术，通过一段时间自动地分析和记录，总结某一个用户特有的习惯和偏好，并将他所希望得到的资讯和应用第一时间呈现给他，方便他的获取。

其次，对于软件及资源提供商，它应是一个用户行为的分析平台。软件及资源提供商所提供的服务将出现在相应软件商店的资源列表中。用户在使用的过程里，系统会自动记录用户的兴趣、使用软件的频率等其他相关数据，并将这些提交给数据分析平台，供软件及资源提供商进行分析，以向用户提供更好的产品。从而真正大幅度提升软件产品销售。

另外，对于硬件设备提供商，软件商店也会作为收集用户体验及产品bug的平台。用户在使用设备的同时，它将会记录用户的操作行为，包括死机、错误提示、意外关机等。在联网的情况下，这些数据就会被传送到服务器，以供厂商进行分析。从而指导其开发出更加优秀的产品。

最后，对于移动运营商，软件商店是一个大的数据中心。平台可以为运营商提供所有相关的数据分析，主要包括用户操作行为、用户使用时间、使用频率、经常使用的软件及功能等，根据这些数据制定业务发展策略，推出有针对性的营销服务。■

选好你的创业切入点

■ 文 / 圣伟

随着移动带宽的发展，以及接近6亿的庞大手机用户群，中国移动互联网未来良好的发展前景，已成大势所趋。正所谓“时势造英雄”，虽然很好的外界形势，为创业者们带来了更多的机会，但是，掌握如何抓住这些机会的方法也是至关重要的环节。

通过对用户需求的了解，我们发现：未来的移动互联网，不应该仅仅存在如“WAP新闻”、“电子读物”、“IM聊天”、“天气预报”等这些大众化的应用。而更需要个性的内容和不同的展现方式，以满足各类用户的需求。那么，选取最合理的技术方向，来打造出一个满足个性应用的手机应用开发平台就是一个很好的创业切入点。

瞄准技术的方向

Widget（微技），是一种基于互联网Web的小应用，通常实现某个特定的功能。微技最初源于苹果电脑的一个插件工具——Konfabulator，现在已经扩展到各种桌面操作系统和手机操作系统上。

Mobile Widget（移动微技）技术，是指运行于移动终端上的微技。微技的应用框架非常适合手机终端，手机终端屏幕相对较小，浏览器却占用了有限的屏幕资源，导致手机上网用户体验较差。而移动微技不仅可以独立于浏览器运行，有效地利用手机屏幕，而且可以更加快速、直接、方便地访问移动互联网。并且，它具有界面丰富、无缝链接网络、开发成本和开发门槛很低等特点，由于基于标准Web技术，所以潜在的开发者也很多。此外，由于运行在移动终端上，移动微技还具有一些其他的特性：

首先，可以通过移动微技实现个性化的用户界面，可以轻而易举让每部手机都变得独一无二；第二，移动微技可以实现很多适合移动场景的应用，如与环境相关、与位置相关的Web应用；第三，移动微技特定的服务和内容使得用户更加容易获得有用信息，减少流量，避免冗余的数据传输带来的额外流量。

现在，iPhone、GPhone、Nokia系列高端手机都内置了Mobile Widget服务。可以预见，今后用户的手机界面中将不再满是各个应用的icon，而是带来不同功能和应用体验的Widget，Mobile Widget正在逐步成为移动互联时代手机的“新型内容门户”。围绕Mobile Widget构筑的新型移动产业链正在悄然形成。

打造个性手机应用开发平台

总体来讲，尽管Mobile Widget目前仍处于起步阶段，并且无论在用户认知、消费习惯还是业务开发、推广、商业模式等方面都还很不成熟，但手机厂商、电信运营商、SP、CP都已经认识到Mobile Widget未来潜在的市场空间，纷纷切入该市场。

2006年，北京易路联动技术有限公司总经理徐国洪就是在瞄准了Mobile Widget的技术发展方向之后，选择了北上中关村进行“二次创业”。由他所带领的五十余人的团队中，80%以上都是研发人员。经过他们两年多的努力，以及过千万的研发投入，他们的产品OpenFace Mobile Widget平台此时已初现端倪。2008年4月，OpenFace Mobile Widget平台和相关开发工具在互联网上正式发布，并且作为开放软件供人免费下载使用，吸引了众多技术爱好者的关注和实践。

由于时下手机操作系统众多，对于手机开发者而言，困难成倍提升。而移动微技恰好能够避免操作系统不同所带来的应用推广障碍，OpenFace打造的理念，就是彻底改变手机内容缺乏的现状，以移动微技技术为依托，主导一个超出操作系统的开发环境，提供脚本语言的简单编程模式（XML+JS），帮助开发者在较短的时间打造出属于自己的手机应用，完成手机内容的个性定制。

开发者可以基于OpenFace平台很容易地开发出很多类不同的小应用，比如：基于Web的互联网应用(网页浏览、在线交互式Web游戏等)；信息呈现(股票信息、天气信息、位置信息、通讯录信息等)；离线小应用(小游戏、小工具等)；移动终端基础应用(短信、彩信、音频播放等)；不但满足自己的平日所需，同时也可迅速在手机用户中快速传播实现增值。■

特色移动应用产品展示

■ 文 / 李洋

UCWEB

UCWEB是一款国产的手机客户端浏览器。它不仅是一款免费的软件，而且符合中国用户的浏览和使用习惯，其内置的网址导航基本上涵盖了中国的各大网站，使得用户可以更轻松地通过点击实现访问，免除了烦琐的网址输入过程。UCWEB也整合了若干个搜索引擎，保证了搜索的质量。围绕用户使用习惯，UCWEB还提供了RSS、网盘和书签同步功能，更方便用户实时获得所需信息。

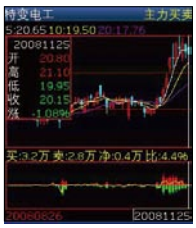
熊猫看书

越来越多的人现在已经把使用手机阅读电子书当成了一种习惯。而熊猫看书就是一款功能强大的手机端电子书阅读软件。它不仅可以阅读TXT、UMD、HTML等格式的电子书，并且还可以查看JPG、BMP、PNG、GIF等多种格式的图像。优秀的用户体验，并且可以自动记忆阅读的进度，使得用户只需简单操作立即体验阅读的乐趣。值得一提的是，配合该公司PC端软件，您可以方便地转换、阅读从网络上下载的或者自己编辑的电子书。

金山词霸-i908手持版产品

- 英汉/汉英双向查询：最基本的功能，使用起来十分方便。当用户的输入不够准确时可以通过模糊查询返回相近词汇的列表。
- 同反义词查询：解释页面含丰富的内容，除基本的音标、词汇解释外，常用单词含有同反义词，并且可以通过点击同反义词，直接获得同反义词的详细解释。
- 屏幕取词：屏幕取词功能让用户可以通过点击解释屏幕界面内任意位置的中文或英文单词或词组得到该词汇的解释，即中英文互译。
- 相关词列表显示等功能：显示查询单词的相关词列表，包含由查询单词组成的词组、相近词汇等。
- 生词本及测试：用户在查询时，可以直接将生词记入生词本，便于以后的学习和使用。同时，可以进行试题的测验。
- 单词真人发音：帮助用户纠正发音及及时查询不认识的单词进行学习。

股票软件： 同花顺手机炒股



手机炒股是最近新兴的一个炒股方式，与传统交易方式相比，手机炒股能做到随时随地查看行情、做交易。 同花顺手机炒股作为一款手机炒股软件，其功能确实丝毫不比PC上的同类产品弱，沪深指数、个股行情、技术分析、个股基本面资料等一应俱全。它还提供大盘、个股最新公告、要闻等各种资讯，使用户无需花费大量的搜索时间就能轻松获悉重要信息。

强大的手机文件管理器：X-plore

X-plore是Lonely Cat Games公司制作的一款功能强大的手机文件管理器。虽然这个本身是一款收费软件，但是却可以无限期免费试用。最新版已经加入了中文语言的支持。这款软件通过树形视图浏览全部的磁盘文件和文件夹，可以对直接复制、重命名或删除任何文件和文件夹，内置对ZIP、RAR和TAR格式压缩文件的解压支持并且轻松创建ZIP压缩文档。并且包含了文本文件、图片和Word文档的查看器，十六进制的编辑器和简单的音频播放器。此外，X-plore可以通过蓝牙或红外直接发送任何文件，也可以直接读取和保存收件箱的文件。

贝多

随着城市的高速发展，很多地方的道路也是越加复杂，如果手里有一幅详细的地图则可以省却很多麻烦。而现在，只要手机里面安装有贝多软件，就相当于有了一幅详细的全国地图。不仅如此，贝多还可以通过GPS做到精确定位，使得你在地图上的位置一览无余，即使没有GPS的手机，贝多也可以通过GPRS进行定位，只是精确性没有GPS那么高。贝多的社区功能也是一大特色，通过地理位置信息，用户可以轻松找到周围使用贝多的其他用户。而这个特色也使得贝多在SNS方面有很大潜力。

可扩展性的艺术（上）

计算环境不断变化，可扩展性（Scalability）方面的要求也越来越高，那么到底什么是可扩展性？原则有哪些？

■ 文 / Haytham ElFadeel　译 / 舒克

在这个系列的第一篇文章中，我将会讨论可扩展性的定义及其原则，接下来我会讨论一些这方面的模式、反模式以及指导方针。现在，让我们开始吧。

可扩展性及其原则

Wikipedia中的“可扩展性”是这样定义的：在电信行业与软件工程中，我们希望系统、网络或是流程能够具备可扩展性，也就是说它们有能力以优雅的方式处理不断增长的待处理工作，或是随时都可以进行扩充。举例来说，某个系统面临着负载不断增加的情况，如果通过增加资源（通常是硬件），系统就能够提升总体的处理能力，我们就说这个系统具备可扩展性。这个词也可以用在商业的上下文中，如果一个公司具备可扩展性，也就是说该公司的业务模式能够为公司内部的经济发展提供支持。

要是上述定义很难理解，不妨看看下面这句话：从最简单的层面来讲，可扩展性，就是要能够更多地去做某些事情。让一个Web应用具备可扩展性，就是希望允许更多人同时使用你的应用。可以说：增加可扩展性，即有能力处理更大数量的并发用户请求。

如今，提升可扩展性有两种方式：

- 纵向扩展性：这是指向同一个逻辑处理单元中加入更多资源以提升处理能力的方式。比如：向应用服务器中添加更多CPU，或是扩展存储设备容量或内存。纵向扩展性仅关注硬件。
- 横向扩展性：这是指为某种资源加入更多逻辑处理单元，并让它们像一个单元一样工作。可以想想下面这些技术：服务器集群、分布式计算、负载均衡。横向扩展性同时关注软件和硬件。

现在问题来了：我们的软件架构应该如何设计才能具备可扩展性？架构应该尽量做到线性的可扩展性，也就是说：随着系统中资源的增加，处理能力应该保持按比例提升。然而，增加资源会导致额外的管理成本，使得难以达到预定目标。Royans K Tharakan称其为“可扩展性因子”，并用其来列举可扩展性的类型：

- 如果在做与可扩展性相关的操作时，可扩展性因子保持为常数。这就叫做线性可扩展性。
- 但很可能有些组件并不像其他组件那么适应规模的增长。小于1.0的扩展性因子叫做次线性扩展性。
- 话说回来，也可能因为增加更多组件而获得更佳的性能（在RAID系统中跨多个磁盘的I/O，磁盘越多，性能越好）。这种叫做超线性扩展性。
- 如果应用程序没有专门为可扩展性而设计，当规模扩大的时候情况可能会变得更糟。这种称为负扩展性。

跟软件开发中的许多事物一样，这里也没有适合一切情形的银弹可以解决你的扩展性问题。如果你急切需要可扩展性，向纵向发展可能是最容易的。但是要注意：纵向扩展性会随着你的规模增长而越来越昂贵，而且无穷的横向线性扩展性只是难以达到，而无穷的纵向扩展性绝不可能。

从另一方面来说，横向扩展性并不要求你购买越来越昂贵的服务器。它的本意是用普通的存储和服务器方案来实现规模伸展。不过横向扩展性也不便宜。在构建应用时，必须从一开始就考虑好底层如何构建，才能保证在多台服务器上运行的应用就像在一台服务器上一样。

构建可扩展软件系统指南

1. 减少处理时间

在任何提到优化的书或课程之中，你会发现：代码级的优化，就是指通过使用更好的算法、代码、设计或是策略来减少处理时间。减少请求处理时间是可扩展性的一个主要指导方针。比如，减少处理一个用户请求的时间，这就意味着你可以在同样的时间内处理更多用户请求。不过，看看下面这些例子：

- 缓存（Caching）：如果数据和代码无法放在一起，应该缓存数据，以减少频繁取数据消耗的各种成本。
- 并行处理（Parallelization）：通过分解问题，并将各个步骤并行处理，就可以降低处理单个工作单位的时间。
- 远程处理（Remoting）：减少访问远程服务的时间，比如，让接口粒度变得更粗。应该记住：远程还是本地，这是两种截然不同的设计决策，一旦确定之后，可不是想改就能改的。还

越来越多网站使用刀片服务器增强可扩展性

要牢记分布式计算的首要原则：不要分散存放你的对象。

- 搭配存放（Collocation）：通过搭配存放数据和代码，可以减少为完成某项工作访问某些数据带来的消耗。
- 使用各种“池”（Polling）：通过各种“池”存放昂贵的资源，可以减少使用这些资源带来的消耗。

我们总是试图引入各种抽象和层次，来让上面的技术变得更加容易。然而软件开发人员总是要么过度抽象、要么抽象程度不足。是的，上面这些概念的确是用来解耦软件组件的好工具，可它们却会使得复杂度增加，从而影响性能。如果你要在各个层次之间进行数据表达的转换，就更是如此。因此，另一种尽可能减少处理时间的方式，就是确保不要抽象过度，同时减少过多的层次。此外，还应该理解运行时服务的成本，不能想当然，除非这些服务有特定的服务水平协议，否则它们很可能成为我们应用的瓶颈。

我的意思是：你不一定要使用上面所有的方式，因为没有万全之策。

2. 分区

减少处理时间还与其他概念有联系，比如我之前提到的代码优化，还有分区、异步处理等等。

通过功能分解或是按功能分区（把相关功能放在一起，不相干功能隔离开），就可以把工作进行分区处理了。进一步来说，不相干功能之间的解耦做得越好，你就能更灵活地对这些功能进行彼此独立的可扩展性处理。比如在许多Web服务器后面有一台单独的数据库服务器，当它成为瓶颈的时候，你必须改变现有的处理方式，方法之一就是采取分区策略。简单来说，你要将单一的架构切分为更小的、更易于管理的小块。将单一元素切分成更小的部分，你就可以横向扩展了。像Amazon、Facebook等大型网站都采用了这样的技术，它们的架构才可以扩展。分区是很好的解决方案。

3. 异步处理

应用分区做完之后，如果想增强可扩展性，就得大胆使用异步处理方式了。如果组件X同步调用组件Z，X和Z就是紧密耦合的。类似的耦合系统只有单一的可扩展性，也就是说要想扩展X，你还必须得扩展Z。在可用性上，你也会遇到同样的问题。想想基本的逻辑：如果由X能够推导出Z，那么非Z一定也能推导出非X。换句话说，如果X不能用了，那么Z也一定歇菜。相比较而言，如果Z和X是通过异步方式整合的，不管是通过队列、多点信息广播、批处理流程，还是什么其他方式，那么彼此都能做到互相独立地进行扩展。而且，Z和X现在就全部具备了独立的可用性特征。即使B当掉了，A仍然可以继续工作，不受影响。

该原则既可以在较高层面应用，也能适用于基础架构层面。类似于阶段化事件驱动架构（Staged Event-Driven Architecture, SEDA）这样的技术，可以用在单独的组件内部完成异步处理，而仍可以保证一个易于理解的编程模型。在组件之间，该原则也是一样的——尽量避免同步耦合。无论是什么事件，两个组件都不能互相直接对话。在任何层面上，都要将流程分拆为多个阶段或步骤，然后用异步方式把它们互相连接起来，这对可扩展性来说至关重要。

4. 并发是可扩展性的精髓

作为一个并行处理的研究者，我可以说：可扩展性的精髓是并发（concurrency），而并发的精髓是并行（parallelism）。并行处理可以减少处理时间，如果工作处理流程是异步方式，效果尤其好。不过要知道，并行可没那么简单。不过下面这些简单的建议可以帮你通过并行减少处理时间。

- 如果你需要保持某些东西（如本地对象、数据库对象等）的锁状态，时间越短越好。
- 尽量减少对贡献资源的争夺，并将争夺共享资源的过程从关键处理路径上拿下来（如用异步方式安排工作）。
- 任何并发设计都应该事先做好，这样就可以深入理解哪些资源可以放心共享、哪里可能成为瓶颈。■

作者简介

Haytham ElFadeel，计算机领域研究者，对于创新、研究、算法和软件开发都有浓厚的兴趣。研究领域包括：语义网搜索、并行计算、分布式系统和高性能计算。目前正在研究语义Mash-up搜索引擎（Semantic Mash-up Search Engine）项目。他的网址是http://www.hfadeel.com/。

■ 责任编辑：郑柯（zhengke@csdn.net）

《开发者面试百问》之参考答案

■ 文 / 朱少民

程序员杂志和InfoQ推出“开发者面试百问”，很受关注。我也快速看了一遍，受其诱惑，选择了“项目管理”和“软件测试”两部分问题做了解答，希望能抛砖引玉，让更多的同仁给出更精彩的答案。

这类题目虽然会考察应聘者的知识面和经验，但不同于学校考试，一般没有标准答案，着重还是考量应聘者的思维方式、创新能力和表达能力。最有名的微软面试题目“井盖为什么是圆的”，就有很多不同的答案，例如：

- 圆的井盖是最安全的，不可能掉下去。如果是方的井盖，在某个角度可能会掉下去；
- 在相同的面积下，圆的井盖所用材料是最省的，通过能力最好；
- 下水井是圆的，所以井盖是圆的；
- 圆的井盖容易滚动，在井盖搬运时，要比方的井盖省很多力；
- 圆的井盖造型是最美的；
- ……

关键是考察应聘者是从哪个角度分析、如何推理演绎、有何新颖观点或思想火花等。所以，下面的解答也许只给出某个侧面，而其他的侧面还有待读者去思考，加深对项目管理、软件测试或软件工程等方面的理解。

项目管理部分：

1. 范围、时间、成本，这三项中哪些是可以由客户控制的？

答：范围、时间、成本，是项目管理中常说的、相互制约的三角关系。任何一方改变都可能牵扯到其他两方的变动。项目管理的本质，就是在保证质量的前提下，寻求这三者之间的最佳平衡。因为客户是需求方和投资方，客户有权对这三者进行控制，希望完成更多的功能，成本降到最低而时间又是允许的，这是最理想的。但同时，它们之间是矛盾的，相互制约的，增加的项目范围，要么增加成本（如增加人手）、要么延长时间。在项目实际工程中，不同的客户关注点是不一样的，但一般更关注项目范围，提出他们的需求——项目要实现的功能特性。然后，根据要完成的工作，和承包方讨论交付时间和费用。承包方，一般应在满足客户的需求情况下，在时间、成本上和客户进行不断交流、谈判。从项目管理的角度看，最好固定其中一项，其他两项可以根据实际状况来调节，最终保证项目质量。

2. 谁该对项目中所要付出的一切做出估算？谁有权设置最后期限？

答：项目的成功是团队协作的结果。在对项目进行估算的时候，也应该让相关人员参与，即由参与项目各个环节的人做出符合实际的估算，最后汇总起来进行综合分析计算，获得项目总的估算结果。项目估算方法有两种方法——从顶向下和自底向上，多数采用“自底向上”。如果采用“自底向上”的方法，项目的最后期限根据估算获得，然后由相关的管理委员会（例如Software Release Steering Committee）来决定。但由于一些客观因素或市场需求，客户、市场部门或企业最高层可能设置了最后交付时间，没有商量余地，这种情况也常常发生。对于这种情况，根据上面决定的日期，实行“从顶向下”的估算，时间确定了，可以调整的是资源和项目范围。

3. 减少交付的次数，或是减少每个交付中的工作量，你喜欢哪种做法？

答：根据项目类型和项目进行中的实际情况来决定。如果项目规模比较大，时间比较长，那就应增加交付次数或者减少每个交付中的工作量，以便及时考察进展，保证项目进度。

在传统的软件项目中，开发周期比较长，往往减少交付的次数，能更好地控制质量；而现在，有部分公司比较倾向于敏捷方法，喜欢减少每个交付中的工作量，交付周期只有几周（最短的周期，可能是1周），拥抱变化，更好、更及时地满足用户需求。而对互联网上的Web应用软件开发，“减少每个交付中的工作量”是比较好的策略，力求及时获得用户的反馈，将用户的需求及时融入新的版本，及时发布出去，赢得竞争市场。例如，在我新书《软件工程导论》中专门有一节讨论“永远的Beta”的软件工程思想。

4. 你喜欢用哪种图来跟踪项目进度？

答：还是要根据项目的特点来决定，具体项目具体对待。当然，有时也不能由自己决定，而是取决于公司已有的工具和习惯。对于复杂、规模大的项目，可能要借助甘特图和网络

图来分析和跟踪进度。简单、规模小的项目，根据进度报告百分比和表格跟踪就可以了。

5. 迭代和增量的区别在哪里？

答：软件开发不是一蹴而就，其过程犹如雕琢工艺品，由无形到有形、由粗到细，很难一次就开发出一个功能完善、强大的版本，而往往是分阶段进行，一个版本接一个版本地发布出去。软件开发分阶段可以通过两种模型来描述，即增量模型和迭代模型。

增量模型描述软件产品的不同阶段按产品所具有的功能进行划分，先开发主要功能或用户最需要功能；然后，随时间推进，不断增加新的辅助功能或次要功能，最终开发出功能强大且全面的、高质量的、稳定的产品。

迭代模型描述软件产品的不同阶段是按产品深度或细化的程度来划分，先将产品整个框架建立起来。在系统初期，已经具有用户所需求的全部功能。然后，随时间推进，不断细化已有的功能或完善已有功能，这个过程好像是一个迭代的过程。最终的目标是一致的，也是为了实现一个强大的、功能完善的、高质量的、稳定的产品。

6. 试着解释一下风险管理中用到的实践。风险该如何管理？

答：风险管理就是通过风险的识别、预测、估算和衡量，选择有效的方法和手段，对风险进行预防、避免、降低或者转移的管理过程。

风险管理的实践很多，包括头脑风暴、风险列表（checklist）等，例如人们常采用十大风险清单。在项目进行中，不时地更新和处理项目当前风险最高的前十项风险，以保证项目不脱离主轨道。因为项目中大大小小的风险会很多，而十大风险清单就是抓住重要的风险并及时处理，一些相关联的小风险可能也就随之消失。从这里可以看出，风险也是动态的，需要经常地、及时地评估当前的风险，例如每周或每两周进行一次风险评估；关键的项目，甚至可能每天做一次评估。

7. 你喜欢任务分解还是滚动式计划？

范围、时间、成本，是项目管理中常说的、相互制约的三角关系。任何一方改变都可能牵扯到其他两方的变动。项目管理的本质，就是在保证质量的前提下，寻求这三者之间的最佳平衡。

答：根据项目特点来定，一般会选用任务分解的计划，责任清楚，可控性更强。滚动式计划的灵活性比较强，适应性比较好，但容易引起大家对计划不够重视，计划能力降低，或者可控性会差些。有时会将这两种方法结合起来使用。

8. 你需要哪些东西帮助你判断项目是否符合时间要求，在预算范围内运作？

答：前提是这个项目的进度计划和成本计划符合项目实际情况，并不断随着项目的时间发展而滚动更新；还要确保收集的进度和花费的成本是真实可靠的；项目的范围还不能影响到时间和成本的规划。满足上述三点，就可以根据项目时间与计划的内容进行对比，来判断项目是否符合时间要求，在预算范围内运作。常用的方法有基线对比法和挣值法。

9. DSDM、Prince2、Scrum，这三者之间有哪些区别？

答：动态系统开发方法（Dynamic System Development method，DSDM）是众多敏捷开发方法中的一种，它倡导以业务为核心，快速而有效地进行系统开发。该方法的详细内容可以参考《DSDM业务中心框架开发方法（第二版）》。这种方法主要是在英国应用比较广泛。一般来说，敏捷方法适合于规模比较小、变化比较快（需求不够稳定）的项目。敏捷开发的方法很多，包括下面所说的Scrum、自适应软件开发（Adaptive Software Development，ASD）、Crystal方法和特性驱动开发（Feature-Driven Development，FDD），可以参考《敏捷软件开发生态系统》。

PRINCE2（PRojects IN Controlled Environments 2）为项目管理提供了一种结构化的方法，这种方法最早是在1989年由英国政府计算机和电信中心（CCTA）开发的，作为英国政府IT项目管理的标准。PRINCE2如今正日益流行，是英国项目管理的标准。这种方法把项目划分为多个管理阶段，保证让所有资源得到有效的控制。依靠严格地监控，项目在控制和组织的方式下得到执行。详细参考：http://zh.wikipedia.org/wiki/PRINCE2。

Scrum（英语的含义是橄榄球里的争球）是一种迭代式增量软件开发过程，通常用于敏捷软件开发，包括了一系列实践和预定义角色的过程骨架，其主要角色包括同项目经理类似的Scrum主管角色——负责维护过程和任务，产品负责人代表利益所有者，开发团队包括了所有开发人员。在每一次冲刺（一般为15到30天周期），开发团队创建可用的、可随时推出的软件一个小版本（增量）。每一个冲刺所要实现的特性来自产品订单（product backlog），产品订单是按照优先级排列的工作需求。在冲刺的过程中，没有人能够变更冲刺订单（sprint backlog），这意味着在一个冲刺过程中需求是被冻结的。管理Scrum过程有很多实施方法，从白板上的即时贴到软件包。Scrum最大的好处是它非常容易学习，而且

应用Scrum不需要太多的投入。详见：http://zh.wikipedia.org/w/index.php?title=Scrum&variant=zh-cn。

过程方法的应用，同样由项目规模、业务特点（需求是否稳定）等决定，而且还受企业文化、流程和领导意识等影响。如果需求稳定、项目规模比较大或周期比较长，一般会选用PRINCE2；相反，可能会选择DSDM或Scrum。后两者没有本质区别，只是具有不同的最佳实践。

10. 如果客户想要的东西太多，你在范围和时间上怎样跟他达成一致呢？

答：首先要向客户说明，如果在某个时间内去做不可能完成的工作，其结果必然是质量得不到保证，或者所花的成本过大。实际上，没有客户想做赔本的生意，每个客户都会重视质量，而不愿意损害自己的利益。

然后，和客户一起，按需求重要性、紧急性等对需求进行分类，分为不同的等级，然后从优先级高的需求开始，来估算不同优先级类别的需求实现的工作量。设定几个不同的开发周期或交付时间，从而由用户做出选择，例如：

只做优先级最高的那类需求，开发周期需要3个月；

做优先级最高和优先级高的那两类需求，开发周期需要7个月；

做优先级最高、高和中等的共3类需求，开发周期需要12个月；

所有需求都实现，开发周期需要18个月。

用户可能选(2)，先完成两类需求，签订合同。等这合同履行很好之后，再继续下一步的开发。

测试部分答案

1. 什么是回归测试？怎样知道新引入的变化没有给现有的功能造成破坏？

增量模型描述软件产品的不同阶段按产品所具有的功能进行划分，先开发主要功能或用户最需要功能；然后，随着时间推进，不断增加新的辅助功能或次要功能，最终开发出一个功能强大且全面的、高质量的、稳定的产品。

答：由于软件修改或变更，对修改后的工作版本所有可能影响的范围进行的测试，就是回归测试。回归测试的目的是发现原来正常的功能特性出现新的问题——回归缺陷，从而确保原来正常的或符合要求的特性，不受其它区域修改的影响。回归测试，伴随着测试过程，单元测试、集成测试和系统测试中，一旦有变更或修正，都要进行相应的回归测试。

通过代码查看或代码评审，可以基本知道新引入的变化是否会给现有的功能造成影响，可以降低回归测试的范围，但没有十分的把握，一般来说，回归测试是免不了。

2. 如果业务层和数据层之间有依赖关系，你该怎么写单元测试？

答：在JAVA中，如果业务层与数据层之间有依赖关系，也就是说业务处理不单纯，这时我们一般用Mock对象来模拟所需要的数据，完成单元测试。简单地说Mock就是模型，模拟测试时需要的对象及测试数据。这类测试工具有MockObjects、Xdoclet、EasyMock、MockCreator、MockEJB、ObjcUnit、jMock等。

不妨再问一个问题：对业务层测试可以用Mock来模拟，而对数据层如何测试？有两种方法：首先可以使用Mock对象来测试DAO。它屏蔽了具体的关系数据库，它的优点是测试代码的编写方便，可以快速运行。缺点：风险太大，对数据层测试的力度太小，屏蔽了很多与数据库相关的问题，比如：对象和数据库表之间映射，查询语句的语法是否正确。

第二种方法是直接在关系数据库中测试。优点：能对数据层进行完整的测试。缺点：单元测试运行速度太慢，要频繁的对数据库进行操作。

3. 你用哪些工具测试代码质量？

答：这就取决于使用工具的经验，而这方面的经验，开发人员更多。适合Java代码的工具有Checkstyle、Findbugs、Jalopy、PMD、Parasoft Jtest、Coverity Prevent for Java；适合C++语言的工具有Parasoft C++Test、Coverity Prevent for C/C++。可参考下列文章：

FindBugs，第1部分：提高代码质量（http://www.ibm.com/developerworks/cn/java/j-findbug1/）

代码检测：Code Review与CheckStyle（http://www.ltesting.net/TestTech/Others/200702/1606.htm）

代码优化：Jalopy（http://www.open-open.com/open32645.htm）

4. 在产品部署之后，你最常碰到的是什么类型的问题？

答：产品部署之后，容易碰到的问题是安装配置不对，测试环境和实际运行环境总是存在差异。其次，出现的问题，可能是系统稳定性问题、性能问题，有可能是由于脏数据、传输中的异常数据和大数据量等引起的。

5. 什么是代码覆盖率？有多少种代码覆盖率？

答：当我们想了解测试是否充分、是否有些地方没被测试过，就需要对所有测试过的地方有所了解，也就是了解测试的覆盖程度。这种程度的量化就是测试覆盖率，即测试覆盖率是

用来衡量测试完成程度或评估测试活动覆盖产品代码的一种量化的结果。它可用于评估测试工作的质量，也是产品代码质量的间接度量方法。如果用公式描述的话，可以看作“测试过程中已验证的区域或集合”和“要求被测试的总的区域或集合”的比值。

基于代码的测试覆盖评测，是对被测试的程序代码语句、代码块、类、函数（方法）、路径或条件的覆盖率分析。如果应用基于代码的覆盖率分析，一般需要借助工具（如IBM Rational PureCoverage、Bullseye Coverage、开源Clover、EMMA、Cobertura和NoUnit等）来执行。可参考下列文章：《代码覆盖率工具Bullseye Coverage》（http://soft6.com/tech/4/46850.html）、《使用emma测量测试覆盖率》（http://blog.csdn.net/KerryZhu/archive/2006/10/13/1333548.aspx）。

6. 功能测试和探索性测试的区别是什么？你怎么对网站进行测试？

答：这个题目本身有问题，把“功能测试”和“探索性测试”一起比较并不合理。功能测试中包含了“按已完成的测试用例或已计划的测试大纲等进行测试”和“探索性测试”，而探索性测试一般也是为了发现功能中的问题，虽然探索性测试还会涉及安全性测试、性能测试等。

功能测试方法中包括等价类划分、边界值分析、因果图、决策表和正交试验法等，也包括错误猜测法，错误猜测法也可归为探索性测试。

探索性测试，也可以称随机测试（ad-hoc test），充分发挥测试人员最大的灵动性、创造性，进行各种猜测和试探，去发现一些相对隐藏比较深或偏僻的软件缺陷。随机（ad-hoc）测试，也可作为一种重要的测试辅助手段，以帮助测试人员尽早地熟悉产品，发现测试用例的不足，添加或改进测试用例。

对网站进行测试时，不仅要做好功能测试，包括功能的逐项验证、针对功能的负面测试、探索性测试等，还要进行安全性测试、性能测试、UI适用性测试等。

7. 测试套件、测试用例、测试计划，这三者之间的区别是什么？你怎么组织测试？

答：测试用例（test case）是为了更有效地发现缺陷而设计的、可以独立地执行的最小测试单元。测试套件（test suite）是为了完成某个测试目标或任务而组织的若干个测试用例的集合。测试计划（test plan）是对测试活动的事先策划，包括确定测试范围、估算测试工作量、识别测试风险、安排资源和进度等。测试计划指导测试用例的设计和测试套件的创建，测试套件是由测试用例构成。测试计划的实施需要借助测试用例、测试套件来实现。

组织测试，简单地说就是：计划测试-> 设计测试用例-> 创建测试套件-> 执行测试套件（转化为执行测试用例）-> 测试结果分析和评估-> 调整测试计划-> 重新优化测试套件和测试用例-> 再执行测试-> …… 详细参见《全程软件测试》。

8. 要对电子商务网站做冒烟测试，你会做哪些类型的测试？

答：冒烟测试（smoke test）这个名称的来历，大概是从电路板测试得来的。因为当电路板做好以后，首先会做加电测试，如果板子没有冒烟再进行其它测试，否则就退回去。软件中的冒烟测试就是在每日构建（daily build）软件包后，对系统的基本功能进行快速测试，以验证基本功能是否能正常运行。如果有问题，就打回开发部门；如果正常运行，说明软件包构建成功，接下来就可以进行常规测试或大规模测试。

对电子商务网站做冒烟测试，包括基本功能测试和性能测试。基本功能测试可以完成一个交易的完整过程，例如从系统登录-> 商品查询-> 选择商品-> 提交订单-> 确认-> 付款-> 结算等。

9. 客户在验收测试中会发现不满意的东西，怎样减少这种情况的发生？

答：客户可能发现功能或界面设计和他预想的不一致、或者会发现有些功能的操作不是很方便、或者发现一些错别字等各种缺陷。针对不同的问题，有相应的一些办法，概括起来就是和客户进行充分沟通，真正理解客户的需求，和客户的理解达成一致。其次，在开发期间，还可以邀请客户参与软件设计规格说明书、测试计划、测试用例等的评审，当软件能基本正常工作时再次邀请客户从头到尾再看一遍（product walk-through）。最后，就是开发人员和测试人员做好自己的本职工作，构建高质量的软件，进行充分的测试。

10. 你去年在测试和质量保证方面学到了哪些东西？

答：通过自己遇到具体的问题来说明。例如某个特定的缺陷分析，使你认识到某个方面的问题，然后找到真正原因，并加以克服。或者通过某个质量事故，增强了“质量第一”的意识，或者由于某些冲突导致项目质量问题，认识到“沟通”、“流程”或“规范”等的重要性。除此之外，还包括从阅读好书、参加培训等活动所获得的收获……■

作者简介

朱少民，国际软件公司资深QA总监，CSTQB资深专家，从事软件工作近二十年，先后出版专著《全程软件测试》和《软件工程导论》，并主编了多本高校精品教材。

■ 责任编辑：郑柯（zhengke@csdn.net）

互联网敏捷开发实践之路

敏捷开发中有很多实践非常适合互联网公司，然而又不能全盘照搬。互联网产品有其自身的特点，必须要量体裁衣，绝不能削足适履。

■ 文 / 王速瑜

互联网行业是一个快鱼吃慢鱼的行业，也是在所有行业中竞争最激烈、变化最快的行业。如何在这样的行业中脱颖而出，敏捷开发理念让互联网公司看到了新的曙光，同时也在互联网产品研发方面掀起了一场革命。

互联网产品研发特点

互联网行业除了竞争激烈之外，其产品服务模式也具有鲜明的特色。互联网的产品往往是面向海量用户的服务，它非常关注用户的行为和反馈，一切以用户价值为核心是互联网产品最核心的特点。这一特点也决定了互联网产品研发具有如下4个关键特性：

1. **产品的高度不确定性**。互联网产品的用户区域分布广泛，用户群体存在不同阶层，行为和习惯也大多不同，因此用户需求的获取充满不确定性。如果没有科学的用户研究方法和体系，没有关注用户行为的量化分析工具和理念，很难给出产品的清晰定位，因此也无法提供真正满足用户需要的产品服务。

2. **产品需要快速响应用户的变化**。因为产品的高度不确定性，大部分的互联网产品服务在整个用户研究、需求分析、产品研发及交付服务的过程中，都采用探索式、适应性的研发理念进行产品的研发。通常，他们会把整个产品研发周期划分为若干个迭代，采用迭代式的演进过程，不断交付新的产品特性，并通过观察用户的反馈和行为，进而随时调整产品的思路和方向。在这样的情况下，时间是一个很好的促进因素。迭代式开发方法让互联网公司可以更好地缩短产品上线时间，并完成产品方向的检验和服务的提供，保证在激烈竞争中立于不败之地。现在大部分互联网公司都有自己的实验室，如Google Labs，在该网站上可以看到Google正在试验中的新产品，用户可以去体验并给这些产品提交反馈，Google的开发团队会根据这些反馈来及时调整产品的方向，以提供更好的服务，这样就形成了一个良好的用户端到开发团队之间的互动，也很好的解决了这些产品本身所存在的高度不确定性的问题。

3. **团队具备高度创新能力**。Google之所以能取得如此巨大的成就，通过对它的研究，我们发现，创新是Google成功的秘宝。Google的创新法则包含精英的团队、创造性的氛围、自由提出想法的渠道和20%自由工作时间的机制。由精英组成的敏捷产品开发小团队作战模式是Google创新得以支撑的最关键因素。

在国内，大部分的互联网公司都还是采用“抄、糙、超”这样的一种产品服务跨越式发展思路，但是实际上，在激烈的竞争环境当中，大部分互联网公司交付的产品都一直停留在“抄”的阶段，而“抄”也更多是形似而神不似。2007年和2008年在Web 2.0的大潮当中，有多少模仿Facebook的SNS社区不断涌起又不断消亡，其本质还是在于缺乏创新，缺乏对用户需求最本质的把握和差异化服务的提供。可见，创新对于互联网产品是多么的重要。

4. **产品的Small Release发布模式**。只有创新和更快的开发还是不够的。互联网公司必须以更快的速度、交付出质量更好的、更符合用户需要的、并且交付成本更低的产品给用户，这样的公司才能具备更好的竞争优势。传统软件公司在产品交付过程中需要耗费大量人力和财力才能解决产品交付问题。而对于互联网产品来说，产品交付有天然的优势，通过互联网这个巨大的平台，产品交付可以以很低的成本交付到用户面前。但是由于有海量用户在使用这些产品，如何保证交付质量，如何保证更好地满足用户需求呢？我们把互联网产品的交付方法定义为Small release。它有两个特点，其一，发布过程是逐步对用户放量的过程；其二是尽量早发布，常发布，注重用户的反馈。通过Small release，我们不仅仅更好地控制了发布过程中

互联网的产品往往是面向海量用户的服务，它非常关注用户的行为和反馈，一切以用户价值为核心是互联网产品最核心的特点。

产品不完善带来大范围影响的风险，同时也进一步提升了产品的质量，并以最快的速度获取到用户反馈，从而有助于改善产品。

总体而言，高度不确定性和团队的高度创新能力，使得互联网产品研发要更关注“用户”和“体验”；快速响应变化和Small Release则说明了互联网产品研发的“适应变化”能力。《敏捷软件开发宣言》所提到的4个核心价值观，它们的要点刚好与互联网产品研发特色相吻合，因此，敏捷开发的理念正不断给互联网公司研发部门所接受，腾讯、Yahoo!、阿里巴巴、校内网等互联网公司都纷纷引入了敏捷开发，并逐步形成各自的最佳实践框架。

敏捷开发实践之路

世上没有轻而易举得来的敏捷，敏捷也不是银弹，你不能指望用简单的3~5步就能实现它。那么，对于互联网产品研发，如何开始敏捷开发的实践之路呢？接下来，我将结合我长时间的实践和思考，通过以下5大步骤来进一步论述。

兴奋参与，平等沟通

对于采用敏捷开发的团队而言，要让敏捷开发能在项目中更好地应用，成员的兴奋度是非常重要的基础。我们提到的兴奋度包括成员对产品的Ownership、成员在工作中的快乐感和自豪感、成员在工作中的学习能力和可塑性等。我们通过在具体团队中推动敏捷开发实践的过程，发现以下的一些点可以帮助团队提升兴奋度。

1. **团队核心人员的示范作用**。产品经理是互联网产品研发的最重要角色。作为团队的核心人员，他的示范作用非常明显。产品经理首先要非常喜欢自己的产品，他需要有把产品当作自己Baby的心态，对产品要体现出热心、爱心、细心和信心，并在团队中进行传播和感染。

关注“用户”及其“体验”，产品要具备“适应变化”能力；这些特点使得互联网产品研发与《敏捷宣言》所提到的4个核心价值观正好吻合。

2. **从用户和市场中寻找积极正面的反馈**。人都有工作自豪感（成就感动机）的追求，因此，产品的不断成功，可以给团队带来极大的兴奋感和成就感。因此，通过定期用户数据和收入数据分析，比如产品同时在线用户数不断攀高、产品本季度盈利再上一个新台阶等等，这些都是很定量的提升团队成就感的方式。

3. **不怕犯错误，培养尝试的勇气**。人都可能犯错误，最优秀的团队不是完全不犯错误的团队，而是善于在错误中总结经验和通过学习不断超越自我的团队。互联网产品研发团队尤其如此，互联网产品的高度不确定性，经常会让我们不断要犯一些“方向性”的错误，我们不能因为犯了错误就去责怪团队、评判团队，而应该拥抱变化、及时发现问题，并进行适应性的改进和学习。

4. **消除沟通障碍和顾虑，保持广泛、平等的沟通**。软件开发活动是一项团队活动，要使这个群体有效运作，成员之间的沟通和交流非常重要。但是我们也发现，很多团队不具备这样的基础，他们在实际沟通过程中存在很多障碍，比如开发人员的内向型心理、沟通方式存在问题等等。因此我们要打破这些障碍，在我的实践中，通过IM工具可以有效解决沟通心理障碍问题，我们会给团队建立一个QQ群，很多想法和讨论都会通过该群来进行，我们发现，在QQ群里面，每个成员都会很放松，很自由，思维也变得比平常更为活跃，它有效帮助我改善了团队的沟通氛围。

5. **保持模糊的工作边界，培养产品的归属感**。传统的项目管理方式强调人员分工，职责清晰化。但是实际上，对于互联网产品的研发，保持一定的模糊工作边界，可以更好地提升每个成员的产品归属感。以产品特性发展过程来说，产品的特性不仅仅来自产品经理的规划、用户的反馈，同样也来源于团队内部。我们看到很多互联网公司都有自己的产品博客，在上面，团队成员可以分享自己对产品发展的看法，提出自己的创意和想法，这些都是提升产品归属感的好方法。如果团队中人人都有机会充当产品经理，有这个产品就是我自己产品的感觉，我们可以想象，产品一定可以获得更良好的发展。

6. **积极寻找公司领导和全员的关注和支持**。不断寻找公司领导的关注，可以保证团队时刻收到激励。特别是邀请到公司老板来参与你的项目时，这种快乐感和自豪感会极大地刺激团队，它能极大提升团队的热情，即使是加班加点也会毫无怨言。在我的实践过程中，我们常常给项目设定阶段里程碑，当里程碑达到的时候，我们会给团队举行一个Party，并寻找老板的奖励，老板会在Party上给团队直接的激励，当然更重要的是，他会给我们的Party买单。

此外，我们经常提到敏捷团队Monkey的角色，一个团队中常常有一个人会充当搞笑Monkey，他每天都会在团队中传递快乐的信息，并以此增强团队的兴奋度，比如每天下午找个时间给团队发一些笑话，让大家在开发过程中也能充满笑声。提升团队兴奋度是解决团队沟通问题的好办法，它作为敏捷开发实践的起步，也许不如敏捷开发所提到的具体实践那么具备可操作性，但是它确实相当重要。它是成功的项目团队所具备的共同特质，它甚至可以极大激发团队的创新能力。

图1 实际开发过程中的故事墙

增强团队内透明度

有观点认为：软件开发是一种交流协作游戏，同时指出：项目进展的速度与信息从一个人的头脑传递到另一个人的头脑所需的时间相关。Alistair Cockburn称其为“信息对流”，他用热气流动来比拟信息在人们之间的流动，并提出“渗透交流”的概念。他认为：在一个物理办公环境，每个人在专注于自己工作的同时，会不由自主地听到一些声音，并能从中不自觉地提炼出一些感兴趣的信息，这可以降低发现和传递信息的成本。因此他觉得物理环境的布局，比如结对编程的座位设置对于信息传递都是非常重要的，同时他提到了在办公局域放置“信息辐射器”的重要性，而信息辐射器所起到的作用就是增强团队之间的透明度。常见的增强透明度的方式有：

1. **故事墙**

故事墙展现项目迭代中的User Story的实现情况，用来透明化项目的实时进展情况。它的目的是使任何人只要进入团队工作区域，就可以立刻了解项目的进度、每个人现在的任务、一些团队需要改进的地方等。

2. **进度报表**

图2是典型项目进展BurnDown Chart的展示图，中文名为燃尽图，透明地展现了项目进展情况。

3. **回顾总结**

回顾总结发生在迭代结束的时候，团队成员针对本次迭代进行总结。它可以透明化地让团队成员都了解到本次迭代做得好的地方以及需要改进的地方。

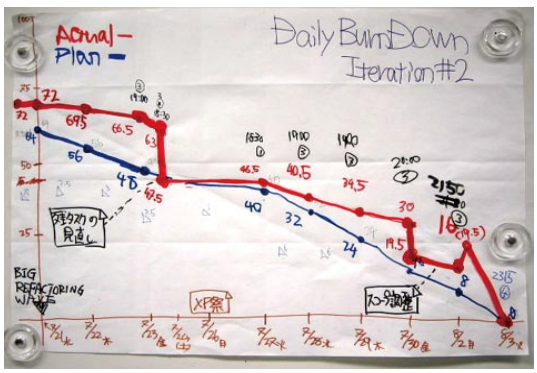


图2 典型项目燃尽图

频繁心跳，张弛有度，保持稳定的迭代节奏

Martin Fowler提到：敏捷过程是基于适应而非预测的过程。他认为，敏捷软件开发是一种自适应过程，通过快速、短迭代式的开发，不断产出和演化出可工作的软件，根据用户的反馈信息做适应性调整，然后进入下一轮快速短迭代式开发。这是敏捷开发最核心的理念之一。

在进行实践敏捷迭代式开发方法时，有2个要点是需要注意的。

1. **稳定的迭代节奏是保证团队可持续交付工作产品的基础。**

一个成熟的团队，总会有约定俗成的团队协作默契。对于敏捷开发团队而言，稳定的迭代节奏可以让产品保持更稳定地交付。我们对一个产品制定了固定的发布周期，比如每周五固定会发布一些产品的特性。这个时候，总会有一群忠实用户来体验新特性，长此以往，慢慢地就培养了用户的习惯，并且也保持了用户对产品的期待。Scurm方法也非常强调稳定的迭代节奏，它把每个Sprint都固定为30天，并以此来保持可控的、稳定的节奏，它假设，30天是人们做回顾总结最好的一个时间段，小于或大于30天都会降低效果。

2. **自适应计划的建立。**敏捷不是不做计划，而是从Plan到Planning！

常常有人刚接触敏捷开发时，产生一个误区：“敏捷是不需要计划的”。说这话的人忘记了一个很明显的现象，敏捷开发小组经常会每隔一到两周就会花大概半天时间来进行会议，并列出任务列表。开发小组会把计划活动扩展到项目整个过程，而不是在项目一开始就先期完成所有计划，但是这样的结果常常被误认为：敏捷是无计划的。

其实，敏捷开发不是不做计划，而是从Plan到Planning，图3就展示了敏捷开发的Planning过程。

在Planning的过程中也有3点要注意：

1. 我们仅仅对下一个迭代做详细计划，长期规划以粗颗粒形式来描述。

2. 整个Planning过程会设定明确的Milestone，Milestone本身也可能发生变化。

3. 到达Milestone的路径是灵活、适应性的。

坚持Small Release法则

前面提到，互联网的产品发布模式是Small Release模式。Small Release使产品的交付一直处在半透明状态，产品上线发布不要一下子就面向用户全体，而是有策略有节奏地逐批放量。它是一种典型的敏捷化发布方式（集市型）。采用Small Release以后，通过逐步放大用户群，一方面让用户参与了产品的“测试”，有利于降低缺陷影响的范围和风险；另一方面也降低了对产品测试资源的投入，没有必要对每一次非正式版的发布进行全面回归测试。另外通过Small Release也实现敏捷原则之一“现场

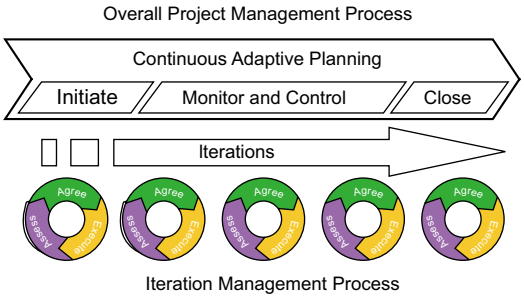


图3 迭代管理过程

客户”，它建立了一种快捷的用户反馈搜集渠道，通过与用户快速交互和收集反馈，可以更好地确定下一次Small Release的目标。业界很多互联网公司都采取这样的产品发布模式。下面是一些例子：

1. 在eBay上，Gmail测试账号曾拍出了200美元，《魔兽世界》的内测账号更拍到了500美元。

2. Napster推出其音乐下载服务的2万个beta测试账户，引来300万的用户注册。

Small Release除了是一种尽早发布的理念外，同时也是一门技术活。实施Small Release 对产品架构有一定要求。大致包括4个方面的要求：

1. 系统切割组件化；

2. 用户规则可配置化；

3. 系统服务版本化；

4. 系统升级平衡化。

同时，在运营环境上，常常需要部署版本服务器，负责对产品不同版本Release的版本控制工作。

加强用户参与，尽快收集用户反馈

对于互联网产品来说，用户体验重要性无需再讲，更重要的是，如何让用户参与？前面提到的Small Release，目的之一就是让用户能尽早参与到产品的体验当中去。此外，我们还有以下方法来让用户更好参与：

1. 直接观察法。直接观察法就是要走到用户实际的环境中去，走入用户的世界，可以采用结构式或非结构式访谈，时间和方式都可以灵活，但是有个原则：第一就是要仔细观察，看看用户真正的工作环境、方式、条件；第二就是要多问为什么。

2. 案例研究法。案例研究法主要是找特定用户群体或典型用户，这种方式不适合普遍的用户目的调查，但是可以作为对边界用户和极端用户场景的分析。这种方法比较适合应用在对现有产品的改造过程。

3. 调查问卷法。最常见的方法，通过设置系列问题来完成对用户的研究。

4. 人物角色法。人物角色法其实是一种数据建模的方法，通过对特定用户数据建模，为每个主要用户群体都创建一个虚构人物角色，然后去捕捉该群体最重要的方面，比如他们试图完成哪些任务，他们的最终目标是什么？

5. 焦点小组（Focus group）。焦点小组依据群体动力学原理请大约6~9个被试（Participant）对某一主题或观念进行深入讨论。焦点小组实施之前，通常需要列出一张清单，包括要讨论的问题及各类数据收集目标。在实施过程中需要一名专业主持人，主持人要在不限制用户自由发表观点和评论的前提下，保持谈论的内容不偏离主题。同时 主持人还要让每个参与者都能积极地参与，避免部分用户主导讨论、部分消极用户较少参与讨论。焦点小组具有群体动力（Group dynamics）、自由开放（Open discussion）、定性数据（Qualitative data）和适合探索目的（Exploratory purposes）等特点。

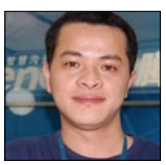
用户参与已经成为互联网产品敏捷研发最关键的一环，它有效弥补了互联网产品由于空间所带来的“非客户现场”感，通过这些科学方法，开发小组跟“客户”又可以保持顺畅的沟通，保证产品是建立在满足用户真实的需求之上的。

一切刚刚开始

现如今敏捷开发已经到了炙手可热的程度，很多公司都已经跟风似地在引入敏捷开发。但敏捷开发真的是银弹吗？是放之四海皆准的真理吗？其实不然，在长期实践过程中，我们发现很多开发组在应用敏捷开发时存在很多误区和形式主义。他们常常以为，每天早上坚持开晨会、产品规划按迭代来计划、定期回顾一下，这样就是在敏捷开发了，然而他们却忽略了敏捷最核心一点——“交付可工作的产品成果”，因此我们都称之为“伪敏捷”，同样还有很多这样的例子。此外，敏捷开发在不同团队的应用也不尽相同，业界也有多种方法体系。但是我们透过这些方法发现，敏捷开发只是一种理念，它传递着以人为本的管理理念，强调团队的自我管理和驱动，并持续改进个人和组织的思想；敏捷开发也提出了一些做事原则，如小步快跑、简单设计，拥抱变化，持续集成等等，这些理念和原则跟互联网产品研发特色异常吻合。

接下来的系列文章中，我将继续分享在互联网企业中，独具特色的敏捷开发实践历程，我会着重提到互联网产品特性驱动的需求管理方法、打造兴奋、平等、参与型团队的项目管理经验、运营形产品的敏捷实践方法等等。我相信，敏捷开发在互联网行业的应用也是刚刚开始，希望通过系列的分享交流，能得到更多朋友的反馈和建议，让敏捷开发能做得更好！■

作者简介



王速瑜，腾讯R&D研发总监，从事产品研发和管理工作，对互联网产品发展趋势，管理理念，技术架构有浓厚的兴趣和深入研究实践。目前主要关注敏捷开发、大规模应用架构、企业SAAS、Web2.0 产品的相关技术和趋势。

■ 责任编辑：郑柯 (zhengke@csdn.net)

以质量为中心的高效软件开发（上）

■ 文 / 高明

引言

种种数据证明：构建高质量的软件有着重要的意义与商业价值。那么那些因素会影响到我们的软件质量呢？举几个常见的例子：

1. 软件的功能越来越丰富复杂，同时软件的各个模块之间的耦合度越来越高。

2. 参与软件开发的人员数量越来越多，技术水平参差不齐。

3. 软件上市日期的限制。

既然我们已经意识到软件质量问题之所在，那么应该如何开发出高质量的软件，同时如何将以保障软件质量作为一条主线贯穿于整个应用开发生命周期当中呢？

不论是传统的瀑布模型（可以看作为一次迭代），还是目前比较流行的迭代模型中，我们都可以将一个完整的开发周期粗略分解为三个阶段，即：编码之前为设计阶段；编码之中为实现阶段；编码之后为验证阶段。软件质量保障就贯穿于这三个阶段之中。做到这些，关键是需要对于软件研发的管理、流程以及方法等多方面的改造或调整，使得“质量为中心”真正的成为我们日常开发的指导原则。一个好的实践方法加上好工具的支持往往能够达到既不会影响研发效率又能够有效提高软件质量的效果。

软件开发周期的各个阶段中，都有一些可以有效保障软件质量的实践方法，下面结合Microsoft Visual Studio Team System 2008（以下简称为VSTS 2008）做一简单介绍。

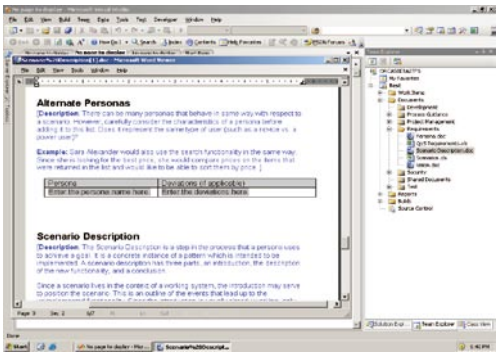
设计阶段——未雨绸缪

很多人认为：软件质量保障，是在软件编码后期或完成以后，由软件测试人员执行一系列的特定测试用例，并运行功能性测试来保证的。到了近期，用于软件编码验证的单元测试逐渐盛行起来，但这些还是不够的。其实，有一些质量保障任务是在代码生成之前就需要进行的。这些任务主要包括了对软件功能需求的早期跟踪处理、早期设计的验证等。完成这些任务就是为软件质量保障做到未雨绸缪。VSTS 2008中内置支持这些任务。

需求与应用场景的有效跟踪与处理

◆ **需求整理**

VSTS 2008内置一系列模板可用于需求或应用场景收集，并且都为Excel或Word格式。这些文档固化在团队项目中，通过对于需求或者首要用户场景的文本描述，即可将应用的特性记录下来并存储在团队服务器Team Foundation Server（后简称TFS）中，供团队成员访问。下图为VSTS 2008的应用场景描述截图。



VSTS 2008中的应用场景描述

◆ **将需求场景映射为具体工作项**

虽然此时需求或应用场景已经有了一个描述，但此时的需求仍然是不能被跟踪，或者说应用需求此刻并没有与具体的开发工作衔接起来。这个时候就需要进行需求与团队工作项之间的映射工作。通过在Excel中直接发布到TFS当中，即可由TFS自动生成一系列的任务工作项。在TFS中，工作项（Work Item）是一种用于反映团队工作的一种单元，或者说工作项也就是一种容器。团队中的所有任务、缺陷、服务质量需求（Qos）、风险、场景等都可以通过工作项来进行描述。同时，TFS使用者还可以定义自己的工作项，用于描述自己需要在开发过程中被追踪的数据。

而对于质量控制来说，我们通常会涉及到缺陷、服务质量需求以及场景这三类工作项。一般来说，缺陷的工作项是在软件开发后期才会出现的，用于记录软件的BUG以及修复记录；服务质量需求工作项是对于类如安全性、运行速度、可扩展性等方面的要求及完成记录；场景工作项即为需求和用例（Use Case）的描述及完成状况记录。

◆ **分配工作项到团队成员**

一旦工作项创建完成之后，就需要把工作项与具体的人联系起来了。这里的具体的人可能就是我们开发过程中各个角色：架构师、开发人员、测试人员等；每个角色可以通过查询自己所需要完成的工作项来知道自己到底

需要完成哪些工作，还有哪些工作没有完成。通过对于工作项的合理分配，团队成员就可以集中精力去解决系统之中要害问题，从而执行在应用开发周期中的质量保障。

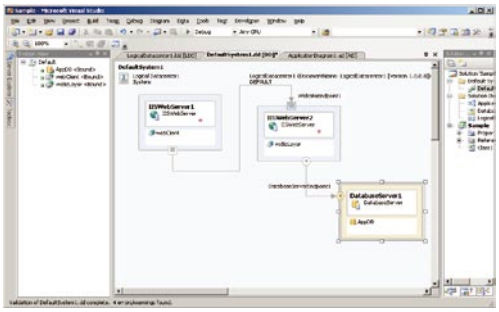
架构验证

软件的架构设计即为将需求（方案和QoS）转化为可构建的设计。一个良好的软件架构应该具有易于理解、易于使用和易于演化的特点；架构设计的优良是软件质量的最首要的保障，但一个架构设计的优良与否是需要验证的，而验证软件的最有效的方式就是通过可执行代码，但往往这个时候是并没有实际的代码，这显然是个矛盾。所以在架构设计的同时如果能够提供一个用于验证的可执行片段，此片段设置用于实现架构方面最重要的方案，同时此片段还能跟现实的设计、代码同步。这将对于架构设计提供有力的检验手段，这也将为软件质量提供有力的保证。在这一点上，VSTS 2008的架构师版本提供了从架构模型生成代码，同时VSTS提供的工具可帮助保持架构工作项、图表和报告与不断发展演进的代码保持同步。而对于架构的验证方面，VSTS 2008的架构师版本能做到下面这些：

利用设计器（Application Designer）通过图表的方式定义应用的架构，并且通过图表将各个组件以及组件之间的分布与通讯状况反映出来。这就让应用架构师能有清晰的架构决策，例如：应用应该用什么样的服务；前端应该看起来是什么样子；并且前端，服务端以及数据库之间是如何通信等。

基础结构架构师（Infrastructure Architect）同样可以利用VSTS 2008架构师版本中的设计器（Logical Data Center Designer）描绘出数据中心的结构，包括不同的服务器之间使用什么服务怎样去会话。同样这些服务器可以包含操作系统版本，补丁级别等信息的定义。

最后，应用架构师将根据部署结构架构来确保应用在已有的基础结构上能够很好的运行。这样能够对于一些潜在的问题就能及早发现，比如：应用的补丁级别或者.Net Framework的版本没有安装在服务器上。下图就是一个应用映射到逻辑拓扑架构图。



应用到逻辑拓扑的架构映射图

实现阶段——深度防御

前面我们介绍了项目经理、架构师、开发人员与测试人员如何使用VSTS2008，以在编码之前通过验证架构保证质量。接下来的工作就是进行源代码的编写了。代码的质量将直接影响到软件的质量，这里VSTS 2008提供了一整套包括：单元测试、静态代码分析、性能分析等工具，旨在改进和保障代码质量。

软件配置管理方面

◆ **源代码分支管理**

在软件开发中，当所需完成的功能特性居多，同时软件开发参与的人员又很多的情况下，可以采用代码分支的方式进行有效的解耦。在VSTS 2008当中，TFS作为源代码配置管理工具，提供了很好代码分支、合并、文件及文件夹的比较等功能，以便快速合并大量文件。使用代码分支，在最大程度上保障了大规模软件的并行开发，同时在最大程度上规避了软件的代码改动混乱。

◆ **搁置集**

在VSTS 2008中引入了搁置集（Shelve）的概念。开发人员在未准备好或者无法签入一组更改时，可以利用搁置集暂时保存所有更改。搁置集可以形象地看作为源代码暂时存放的书架。其典型应用场景典型的为：中断，当所具有的更改未准备好签入但需要从事其他任务时，可以搁置这些更改以保留它们；集成，当所具有的更改未准备好签入但需要与其他团队成员共享这些更改时，可以搁置这些更改并让其他团队成员对它们取消搁置；评审，当所具有的更改已准备好签入并且必须经过代码评审时，可以搁置这些更改并通知该搁置集的代码审阅者；备份，当正在做的工作要执行备份但未准备好签入时，可以搁置已做的更改并将其保留在TFS上；移交，当正在做的工作要由其他团队成员完成时，可以搁置已做的更改以便更容易地进行移交。

◆ **签入策略控制**

在VSTS 2008中，对于源代码的签入可以通过一些签入策略（Check-in Policy）来进行限制和控制，签入策略旨在通过对签入操作和签入文件的检查来确保代码质量。这些策略可以是类如：源代码的签入必须与相关的工作项关联（如：源代码的签入是与某个BUG工作项关联）；源代码在签入之前必须经过单元测试；源代码在签入之前必须进行了静态代码分析等；此外签入策略是可以进行自定义的，TFS使用者完全可以根据需要定义自己的签入策略以保证签入代码的纯净程度。■

作者简介

高明，微软开发平台合作部开发合作技术经理。具有十多年的IT专业技术经验。历任程序员、项目经理、技术讲师、技术顾问等。于2006年加入微软公司。目前负责微软开发技术、开发工具以及VC++技术的推广。

■ 责任编辑：郑柯 (zhengke@csdn.net)

用户界面检视法新探：假用户 CROSSOVER 真砖家

■ 文 / 何立

背景

作为用户体验团队中的一名用户研究员，笔者在实践中发现，某些设计问题的确是无需用户测试验证便能识别的，但设计师可能会质疑用户研究员在交互、视觉设计方面的专业性。那么，站在用户研究这个角色的立场上，怎样能在用户测试前较有效地把UI存在的问题传达给设计师呢？无疑，一个客观合理的评估方法会有助于保证对UI的分析和反馈是具有建设性的。同时，从成本和效率的角度考虑，该方法应该是简单易操作、并适用于整个用户体验团队的成员去快速发现并解决问题的。

方法论简介

1. 理论基础

在可用性研究的理论框架中，包括测试、探寻及检视三种方法论（图1）。其中，检视法是不涉及真实用户的，常被称为专家评审法。本文所述方法属于检视法。所谓“假用户”，源自对认知过程走查法的简化，指模拟用户体验使用UI；“真砖家”则源自启发式评审法，指使用经验准则等评审UI。“假用户CROSSOVER真砖家”，即为一种角色代入式的UI评估法。

正如前文所述，笔者希望探讨相对易懂易操作的手段，因此把原本严谨复杂的检视法简化。而通过分别代入用户和砖家两个角色，一方面是为了尝试贴近真实用户的心智模型，另一方面是为了系统、详尽地对一个UI进行分析。砖家这个角色要借助可用

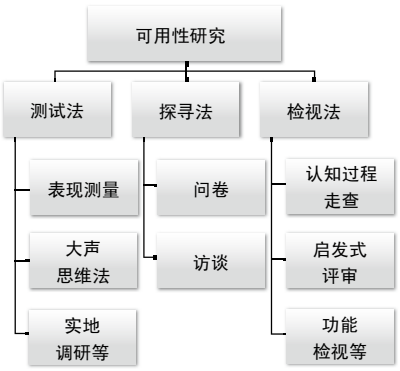


图 1

性经验准则（即一些设计理念、原则）、要对产品概况，如用户群、产品目标、技术约束等有所了解，还要具备一定的用户体验领域的知识。至于为何名曰“砖”家？原因有二：

- 真正名副其实的专家，需在业内有长期的沉淀和经验积累；
- 降低门槛，让不是专家的人也能做出积极、有建设性的评估。

2.方法论的目标

- 提出系统客观的评估分析意见，从而探索解决方案。每个UI都是设计师的心血结晶，“找茬”者应用客观的态度和方法来找出并解决问题。本评估法最重要的目标就是提供分析的切入点。
- 快速改善设计。作为一种时间、金钱成本最低的评估法，该方法能保证在缺乏用户测试支持时能尽可能挖掘问题，且力求将评估过程简化，以实现高效的迭代设计。
- 完善用户测试所用原型的质量。不少问题可在用户测试前被解决，无需用户来验证。

实际操作

1. 时机

越早使用该方法对UI进行评估越好，并鼓励迭代操作（原型阶段、视觉化阶段、开发阶段）。

2. 对象

用户体验团队的任何一个人，都可以既当假用户，也当真砖家去评估其他设计师的作品。而设计师可邀请用户体验团队以外的人或团队里的同事充当你的假用户，因为自己较难投入地充当自己作品的用户。但可代入砖家的角色自评。

3. 过程及内容

3.1 作为假用户

该过程有两个步骤：

1）安排若干相关任务并使用UI完成任务。无论是邀请他人还是自主去做假用户，可安排若干该UI皆在帮助用户完成的任务（主要功能点）。目的是更自然更投入地使用这个UI，避免一开始便带着职业眼光去挑问题。

2）在完成任务后，思考以下四点：

- UI是否易于理解和操作？
- UI是否符合使用习惯？
- UI是否易导致出错？
- UI是否有吸引力？

问题的答案如同一些“信号”，帮助你在第二阶段的分析过程中找出相应切入点。

3.2 作为真砖家

该过程有三个切入点：交互设计、视觉设计和文案撰写。

1）交互设计

对交互设计的评估包括交互控件

与交互过程两部分。控件是各个零散的元素；将控件组织起来，引起交互行为，形成交互事件，即为交互过程。

对交互控件的评估是指审视是否正确选择并使用某控件。正确是指：

- 符合控件使用规范。在雅虎设计模式库中（图2），科学严谨地对每个模式该何时使用、如何使用才能发挥其设计意义进行了详细说明，并辅以相应的理由，值得参考。

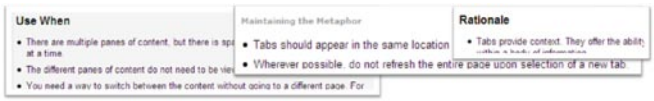


图 2

- 符合使用情景（context）。即使控件符合其使用规范，但用在了不恰当的使用场景中，也是不妥的。
- 前端开发性价比高。对交互过程的评估可考虑三大方面。
- Bug，即是否存在技术性硬伤。
- 导航。导航可以是网站整体级别的，也可以细微到一个特定的交互过程中。在评估导航时可关注：

√ 是否提示任务进度、状态？

√ 是否良好支持前进、返回？若存在大量对后退键的依赖，则说明这方面做得不足。

√ 当前交互过程结束后的引导。举例来讲，注册流程是否到注册成功就结束呢？对于新用户，很多数据皆为空，所以流程结束后，不应把用户悬置，还应考虑下一步该将他们导向何处。

- 交互事件。可从图3所列举的四方面评估交互事件。这四方面是递进的：可用性是基础，效率是在保证此基础后的进一步要求，而反馈、一致性能更好地提升用户满意度。须指出的是，可用性问题往往更常在用户测试中暴露。原因后面会述及。而砖家的价值在于对后三方面的评估，因为这往往是普通用户在测试中较难直接指出的。

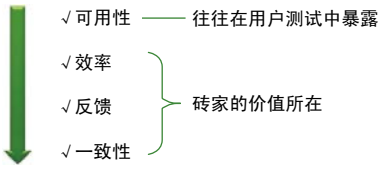


图 3

√ 可用性：基础任务是否能完成？

√ 效率：

①认知负担是否繁重？即用户是否需要思考良久才操作，或操作过程中反复停下来思考？

②鼠标操作、视线追踪是否迂回曲折（图4）？可通过画出可能的操作流程箭头图来进行评估。

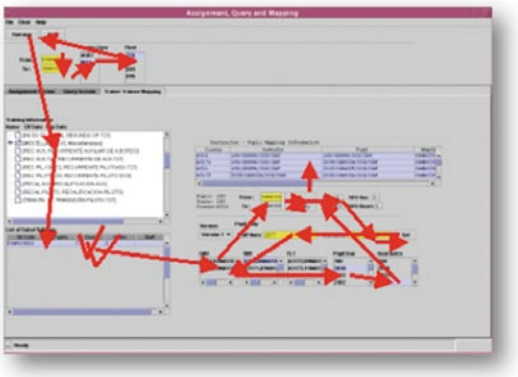


图 4

③是否容易造成误操作？即使操作流不复杂，还应进一步考虑是否易导致误操作。

④是否已考虑极端使用情景？要考虑数据量增大时对效率的影响，以及是否提供相应的检索工具

√ 反馈：

①时机：是否需要反馈？反馈何时出现？

②强度：视觉、文案是否有效、恰当地传达系统信息？

√ 一致性：交互上的一致性是指，一些相似交互事件的交互过程（如步骤、反馈等）应保持一致。

2）视觉

本方法中对视觉的评估并非针对美感、风格等美学元素，而是以格式塔原理为基础，尝试对视觉进行客观分析。内容包括：

- 亲密性：

√ 相关元素是否组织在一起？

√ 不相关元素是否存在可能误导用户的亲密性?

√ 是否有太多孤立的元素？

- 对比：完全不相同的元素是否截然不同？
- 对齐：

√ UI元素是否整齐有序？

√ 对齐是否统一、有条理？

- 重复：元素是否有一致性？

3） 文案

对文案的评估包括用词和行文。用词是指分析：

- 用词是否保持一致性？即同一概念在不同情境下是否使用统一词汇进行表达。
- 是否使用用户的词汇？

行文则是分析：

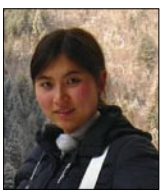
- 是否清晰无歧义？
- 是否存在冗余信息？

能否代替可用性测试？

答案是否定的。原因有三：

- 作为经常接触互联网产品的设计师，操作水平较高，而用户更多是普通互联网用户。
- 对产品或网站整体过于了解，影响对基础问题的判断。
- 不了解目标用户的使用情景。以笔者工作的电子商务公司为例，用户群体各异。除了买卖家之分，买卖家群体也有不同类型的细分。■

作者简介



何立，美国北卡罗莱纳大学人机交互硕士专业。现为淘宝网用户体验设计部，用户研究组的用户体验分析师。

一分钟先生

邮件收发 123

邮件没有称呼、正文、署名，内文密密麻麻、没有排版，附件忘记添加……当你无可避免地阅读到这样的电子邮件时，是否开始怀疑它的沟通效果？但不规范的电子邮件却在日常工作中大量出现、惹人烦恼。那么，如何写一封有效的商务电子邮件，在写作中又有哪些问题需要注意呢？

林锐　上海漫索计算机科技有限公司总经理

致某软件学院工程硕士班的公开信

我授课的某软件硕士班，竟然有40%以上学生发给我的E-mail（交作业）没有标题，或没有称呼，或没有署名，或者全部空白。

你们都大学毕业了，连E-mail都不会写。这样没头没尾的E-mail，不是简洁，而是无知愚蠢，只能当垃圾E-mail处理。

我在课堂上多次强调过E-mail的重要性，Email能够反映一个人是否认真、真诚、尊重别人，而且还拿E-mail案例讲解过如何写一封规范的电子邮件。

我每收到你们的E-mail，就生气一次，一边生气一边改作业，还得让大部分人拿到学分，我已经气得忍受不了必须骂人了。

你们怎么对得起我的授课，怎么对得起自己的学历，怎么对得起父母对你们的期望。

我花费自己的时间写E-mail批评你们一通，那是很给面子了，因为我好歹还有“恨铁不成钢”的愿望。

希望你们永远不要发白痴E-mail了。

邵荣　群硕软件中国区资深技术总监

每天要处理数百个邮件，对于如何更好利用电子邮件这个工具，我有一些经验和建议：

1.带有目的性的沟通

在邮件的开头就要直接说明这个邮件需要表达的意思：“……这个是一个项目状态更新……”；“……请于周三前发出执行计划……”。

2.将邮件作为记录、确认、通知的工具

与电话、面对面沟通比较起来，邮件缺失了人和人之间的情绪感知，所以比较容易产生歧义，经常看到邮件来回很多轮来讨论某件事情。更好的做法是通过电话或者面对面的沟通，将一些可能有争论的事情确定下来，然后通过邮件做一个记录和确认。

3.更精确的沟通

有些发出来的邮件经常会缺失必要的一些信息。例如“……我们在使用产品的过程中碰到了一些问题……”，不如直接说明“……碰到了三个问题如下……”。每次发出邮件前，想想收件人是否会有疑问，然后在第一次写邮件时候就把邮件讲完整、讲清楚，不用等对方再回复邮件来询问或者猜测。

还有一些小建议，例如选择性地使用附件；谨慎使用BCC-密件抄送等等。要记住，电子邮件只是一种沟通的工具，应该关注怎样不断提高沟通过程的质量，而不是过度依赖于工具本身。

关注IT人自我管理、职业发展，广邀经验人士排疑解惑。
花一分钟阅读，还你一个明白。
有难题吗？一分钟先生回答你！请访问http://subject.csdn.net/onemin

感谢北京博文视点资讯有限公司武汉分部对本期“一分钟先生”栏目的协助和支持。

张志　湖北新蓝海网络科技有限公司总经理

http://blog.vsharing.com/hopeful/

令人头痛的邮件有三类：

1.垃圾邮件。对付它们，要选择筛选功能好的邮箱，例如Gmail和QQ邮件，这样可以屏蔽掉大量垃圾邮件。

2.不规范的邮件。例如邮件没有好的主题描述、正文排版不合理、邮件字体过小，邮件背景过花，这些邮件作者只考虑自己的想法，没有站在邮件被浏览的角度考虑，增加了阅读者的工作难度。

3.粗心的邮件。写了正文掉了附件，不得不再次催促。或者是附件过大，没有考虑接收人是企业邮箱，一般不能接受大附件，导致发送失败。

因此我建议写邮件注意五点：

1.尽量选择一个好邮箱，需要来回讨论的邮件用Gmail相互沟通特别合适，历史沟通记录一目了然。

2.养成认真设计主题的习惯，好的主题要有称呼、邮件内容说明和发送者姓名，便于阅读者确认。

3.邮件正文要么用简洁的模板，要么不用任何模板，排版整齐，大小恰当，便于别人阅读。

4.如果有附件一定要仔细检查附件是否添加，如果附件有几个文件，建议压缩打包便于他人下载。

5.邮件发送后设置一个要求回执功能，重要的邮件一定要短信或电话通知接受者。

阿朱　北京润霖汽车科技公司技术总监，著有《走出软件作坊》

http://blog.csdn.net/david_lv

我们公司经常分散办公，最大的问题就是信息隔阂。为了缓解这个问题，我们一般都将邮件进行主送与抄送。为了有效管理众多邮件，我们制定了这样的方法：

1．全公司统一邮件客户端软件。各部门统一设立邮件规则并实施；有专人维护公司全部联系人列表，有更新即发布，方便大家导入更新；有公司级别的邮箱，该邮箱发出的邮件每个人都能收到。

2．在邮件客户端软件中，按部门、关键人、客户，分别设置规则目录。

3．对于项目或固定长期业务，会专门在邮件服务器上建立邮件组，这样发送与抄送的时候就不会有遗漏。

4．为项目或固定长期业务专门设立邮件目录，这样该项目或该业务的全部邮件都集中在一起，容易按日期排列梳理来龙去脉，也容易快速搜索。

5．对于计划事务，我们会发送日程安排，到时会自动提醒。

我个人还有五条心得：

1．邮件主题最好注明关于什么人什么事，概括邮件内容，便于查找。

2．对于有后续事情处理怕遗忘的邮件，可以设立后续处理计划和提醒时间。

3．不要频繁查看与处理邮件，最好固定间隔时间来收发，否则很容易得“邮件焦躁症”。

4．如果有紧急事务，并希望通过邮件的方式留下来龙去脉的信息，可以发完邮件后，给对方发一个MSN消息。如果对方在你可接受的时间范围内没有反馈，可以电话与对方确认。

5．把工作事务邮箱就作为工作用，不要用工作邮箱来注册各个网站或网上信息服务，很容易收到很多广告和SPAM邮件，把工作事务和其他事务混杂在一起。

解析实施功能测试工具的误区

功能测试工具的确可以提高工作效率、提升工作质量、降低开发成本，然而其实际实施过程中却有众多误区和困难。

■ 文 / 吴海荣

随着中国软件业的不断发展，越来越多应用软件开发企业开始关注应用软件的功能测试工具。通过功能测试工具的实施，各软件企业希望通过提高功能测试的自动化水平，达到节省人力资源、提高工作效率的目的。然而在具体的实施过程中，他们往往会遇到很大的困难和阻力，特别是在一些大、中型项目上的应用，更是困难重重。就目前而言，能做到完整的部署、用好功能测试工具的企业并不多，显示了功能测试工具在实施过程中的难度。实施者对功能测试工具理解上的误区是造成工具实施困难的重要原因，如何认识这些误区、以及由此引发的一系列困难，并使困难得以解决，显得尤为迫切。本文试图探讨功能测试工具实施过程中的一些理解误区及由此造成的一些实施困难。

功能测试工具的发展历程及作用

首先我们认识一下功能测试工具的发展历程。自动化功能测试的萌芽，是将测试人员的动作进行简单的录制，并在适当时候回放从而达到自动测试的目的；而后发展了自动化的脚本技术，通过录制过程形成一些可以执行的脚本，运行一些既定的测试案例。这些脚本还可根据不同的测试需要进行编辑：为了解决在自动化脚本重用性差的问题，产生了数据驱动模式，即将数据与脚本分离，测试人员可以在一定范围下调整测试数据，以期达到不同的测试目的，这使自动化工具的适应面更加广泛；为了让更多的业务测试人员能掌握和使用自动化测试技术，产生了一种业务驱动模式的自动化脚本编制技术，业务人员通过简单而严格的业务语言编写测试案例，通过工具解析这些案例，自动形成脚本的雏形，再对脚本进行修改和完善，形成严谨的自动化测试脚本。

功能测试工具因其固有的特性决定了其特定的作用，它可以在大规模回归测试当中运用，可以深入拓展原有手工测试的范围，可以在一定程度上充分利用现有测试资源开展强度更大的测试工作。其具体作用如下：

1. 对项目周期中不断构建的软件版本进行回归测试。这是工具最主要的任务，也是工具运用最广泛、最有成效的地方。

2. 对不同基础平台（硬件平台、系统软件平台）上应用软件的测试。

3. 功能测试工具可以执行一些手工测试困难或不可能完成的测试，例如一些需要连续做海量的业务数据，方可进行业务功能的验证测试。

4. 功能测试工具可以在夜间、在多台虚拟机上运行，为有效调配测试资源提供了可能。

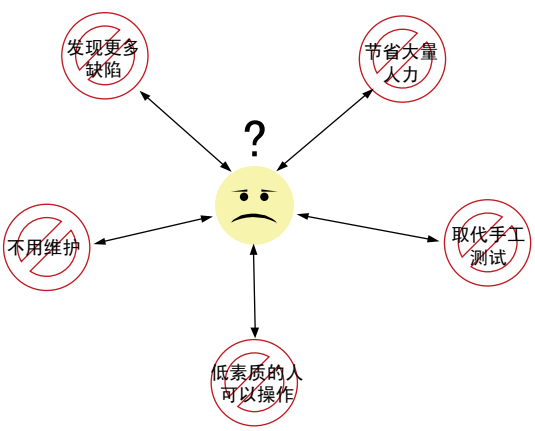


图1 功能测试工具常见理解误区

功能测试工具的理解误区

功能测试工具的实施企业都很了解功能测试工具带来的好处，然而多数软件企业是高估、或者夸大了其功效，导致最终功能测试工具实施的失败。以下是常见的一些理解误区：

误区一：实施功能测试工具可以发现更多缺陷

大多数功能测试工具实施人员认为：使用功能测试工具，可以大幅度提升软件项目在版本发布前的缺陷清除率，可以发现很多手工测试无法发现的缺

陷。功能测试工具只是测试人员使用的一个工具而已，其核心原理是通过自动运行的脚本，模拟人工测试，并没有人工智能等复杂的技术含量。其既不会像人一样会衍生思考，也不会像人一样会沟通协调，其只能简单的执行测试人员规定的测试步骤。因此希望通过实施此功能测试工具发现更多的缺陷，是不切合实际的。在人工智能技术达到一定程度可能可以实现此梦想。

误区二：实施功能测试工具节省大量人力，可以迅速降低成本

有些软件企业认为当测试人员严重不足时，可以通过功能测试工具的实施缓解人力资源的紧张情况。其实大多数情况下恰恰相反，测试人员的工作量在短期内反而会增加。

首先，测试人员需要了解功能测试工具的全面部署情况，需要一定时间熟悉、掌握功能测试工具；其次，需要进行自动化脚本的设计、开发工作；编制好脚本后，他们需要合理规划自动化脚本的自动运行，需要耗费时间核实自动化脚本的运行结果；最后，随着软件版本变更，测试人员还得不断维护这些自动化脚本。除此之外，手工测试的多数工作仍然需要保留，例如编制手工的测试案例（即使是自动化脚本也同样需要编制手工的测试案例或者测试步骤）。因此，实施功能测试工具一定程度上会带来额外的工作量。

误区三：功能测试工具可以完全取代手工测试

有些软件企业认为，功能测试工具可以大量代替手工测试，甚至在不久的将来可以完全取代手工测试。抱有这种想法的人多半是低估了测试工作的复杂性和高智慧性，认为测试是一项任何人都可以进行的工作。其实一个好的测试工程师是一般工具无法模拟和替代的。有些性质的测试工作，手工测试比使用工具来得更便捷：例如核对打印机打印的内容是否符合设计的打印格式，验证本系统外的其他

功能测试工具无法完全取代手工测试，一个优秀的测试工程师是一般工具无法模拟和替代的，而且工具也很难判断有些人工方式可以简单核对的内容。

系统的运行结果是否正确，比对一些字符的大小、颜色是否与设计的一致等等。这些通过人工方式可以简单核对的一些测试内容，使用功能测试工具来判断难度较大，耗费的成本也非常大，使用功能测试工具来实现此项工作是“不经济”的。因此，即使实施了功能测试工具，仍要保留很大一部分的手工工作。

误区四：自动化脚本建立后即可一劳永逸，无需再进行自动化脚本的维护

很多人都认为自动化脚本一经设计、开发就可以随时使用，不用维护了。其实功能测试工具实施的一项庞大工作就是不停维护这些自动化脚本，缺乏维护的自动化脚本是一堆无用的摆设。管理、维护自动化测试脚本必须与软件项目版本的维护同步。软件版本发生变更的时候，测试人员需要比对哪些自动化脚本可能受到影响，哪些脚本可以继续使用，哪些脚本需要及时修改。否则自动化脚本将因无法适应软件变动而变得无法使用。这样编制自动化脚本未能实现多次重用，也就不能发挥自动化脚本的最大经济效应。

误区五：较低素质的人可以操作执行功能测试工具

很多实施者认为自动化脚本建立之后可以交给一些素质较低的人员进行使用，因为最复杂的脚本设计和开发工作已经完成了。其实这也是在一定程度上陷入了“测试是一项简单劳动”的理解误区。首先，运行自动化脚本的人员需要熟悉自动化脚本的具体功能；其次，自动化脚本一旦发现软件版本的缺陷时，脚本的运行人员需要通过报错信息定位软件版本程序的缺陷；最后，运行人员需要根据脚本运行的情况，不断调优自动化脚本，使其发挥最大效应。因此不熟悉自动化脚本的结构和功能，很难运用自如，不具备一定的脚本编制能力和测试技能就无法成为一个合格的功能测试工具使用者。

实施功能测试工具的常见困难

回顾完一些认识误区之后，下面探讨一下由此引发的困难，这些困难在功能测试工具的实施过程中将经常遇到，也是导致功能测试工具实施夭折或者名存实亡的主要原因。

1．组织问题

功能测试工具实施是一项系统工程，需要具备多方面的因素方可完成。其中最重要是人才组织，工具实施团队需要由一批综合性人才构成。首先，实施团队中要有人懂得应用软件测试，若是一批从未做过测试的人来实施功能测试工具，遇到的阻力和付出的代价无疑是巨大的；其次，实施团队中要有人对功能测试工具非常熟悉，一般而言，这些人热衷于自动化测试，熟悉工具的特性，并能熟练运用工具的各种功能；第三，要有能对工具的实施进行整体规划和准确定位的人，这些人在团队中起带动作用，能循序渐进让更多的人了解并熟悉功能测试工具，最终让工具得以广泛运用。然而，在工具实践过程中这一看似简单的组织问题却常常被忽视，团队成员总是无法面面俱到，总是组织准备不充分就仓促实施功能测试工具，导致实施工程的最终失败。

2．自动化脚本与测试数据的平衡关系问题

应用软件的测试数据比系统软件的测试数据更为复杂和繁琐，这也导致了应用软件的自动化脚本较难做到数据与脚本的完全分离。要么牺牲数据的可维护性，要么牺牲数据与脚本的关联性。因此工具实施者容易走向两个极端，一种是为了建立数据和脚本的紧密关联，使得数据可维护性小，也即脚本本身适应能力差，导致实用性差；另一种是数据和脚本过分分离，脚本的执行过分依赖数据的全面设定，每次执行脚本都必须维护一次数据，导致脚本的重用性差。若无法合理解决这对矛盾，功能测试工具的实施将陷入泥沼当中。

3．自动化脚本管理、维护混乱的问题

很多功能测试工具在实施后，没有一整套完善的制度、方法，对建立的自动化脚本进行管理和维护，导致管理混乱。自动化脚本建立之后，迫切需要建立与软件项目同样的版本维护机制，建立自动化脚本与软件项目版本之间的关联。软件项目版本变动可知自动化脚本是否需要变化，可知哪些自动化脚本需要运行。没有持续维护的脚本是无法顺畅运行的脚本，是无法发挥测试效应的脚本，这些脚本只会浪费测试人员的时间，增加测试人员的工作量。因此若没有严格的项目版本管理，则无法进行功能测试脚本的版本管理，也无法进行功能测试工具的后续实施工作。

4．众多不现实的期望影响功能测试工具的实施进程

实施功能测试工具的团队总是希望通过功能测试工具实现很多不可能完成的测试任务，希望大大提高工作效率，希望大量降低人力成本，希望发现更多缺陷，但是这些愿望都可能会在初期阶段妨碍、限制功能测试工具的实施。先让功能测试工具走向正规、实现真正可用，再去考虑更高层次的要求可能会更加现实一些。特别初期阶段需要克服一些不切实际、不着边际的期望，如功能测试工具替代人工测试的一切工作或者大部分工作。切入功能测试工具使自动化测试工具走向正常的运行轨迹是工具实施的难点，在度过这个难点之后再考虑对功能测试工具的其它要求是稳健的实施策略。

如何有效实施功能测试工具

以上分析了功能测试工具实施过程中的几点误区和实施的主要困难，那么如何才能在应用软件开发过程中有效实施功能测试工具呢？其实只要有序的按照一定的原则和策略，功能测试工具也是可以较顺畅实施的。在此对功能测试工具实施过程中的主要注意事项做了以下归纳和总结：

1．纠正思想误区，准确定位功能测试工具

在功能测试工具实施之前，首先必须对功能测试工具有一个全新的认识和定位，认识到功能测试工具能够做什么，不能够做什么，消除对功能测试工具的一些误解。特别是工具实施的初期阶段应尽量降低对功能测试工具的期望值，并制定一个循序渐进的实施计划和实施方案，确定一个初期、切合实际的实施目标。

2．做好实施团队的成员组织工作

功能测试工具的实施团队需要具备各方面素质的成员参与进来，实施团队成员必须非常了解功能测试的方法和理论；实施团队需要掌握功能测试工具的具体功能，工具能够实现哪些功能，不能够实现哪些功能；实施团队的核心成员非常清楚现有的开发流程、测试流程及相关的管理制度和办法。若实施团队无人了解原有的流程和制度可能会处处碰壁，最终实施团队会怨天尤人，认为绝大多数使用者对功能测试工具有强烈抵触心理。因此实施团队核心人物的整体素质和团队的成员构成都是非常重要的。

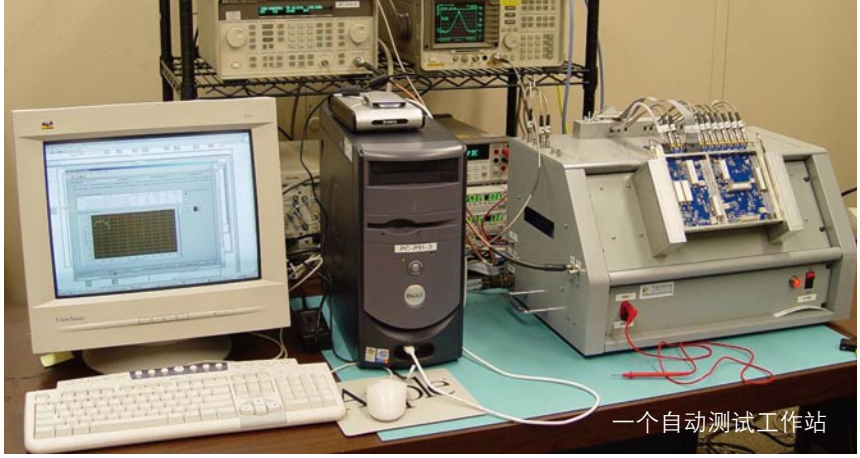
3．熟悉并改进现有的项目管理、测试管理的流程和制度

由于在功能测试工具实施过程中，必然要对现有的工作流程、工作方法造成冲击，其在一定程度上需要适应原有制度或者原有制度做适当变更。例如功能测试工具实施前，测试团队只需要维护手工测试案例就可以了，且手工的测试案例严谨性缺乏硬性制约。当应用程序发生变更时，手工测试案例常常变更滞后，甚至不变更。但是实施功能测试工具后，自动化测试脚本则必须随着应用程序的变更而变更，否则这些自动化脚本将无法继续使用。如何继续管理、维护这些手工测试案例、自动化测试脚本，成为实施功能测试工具后必须面临的问题。

在测试管理流程中，可以增加自动化测试所涉及角色（自动化测试脚本的设计开发人员、脚本执行人员、脚本管理员）的定义，并在测试管理流程各个阶段设定每个角色应承担的具体工作。下面的具体工作安排可供大家参考。

在项目设计阶段，由脚本设计人员进行自动化脚本的设计工作；在项目编码阶段，由脚本开发人员进行自动化测试脚本的编制工作；在功能测试开始前，由脚本执行人员运行部分关键路径的自动化测试脚本，进行版本提交后的冒烟测试；在各功能测试轮次之间，由脚本执行人员进行自动化回归测试工作；在版本发布前，全

准确定位、做好组织、改进流程、找好切入点、建立管理机制、设计策略，把这六点做好，就能有效实施功能测试工具。



一个自动测试工作站

面运行自动化测试脚本，保证版本打包的正确性；在项目结束后，由脚本管理员完成项目涉及各个应用最新全量版自动化脚本的整理和归档工作，以方便下一版本的使用。上述流程的改造，将自动化测试工作定义为测试管理的一项例行工作，严格定义了自动化脚本的编制、运行、维护的各个环节，这也为自动化测试工作的实施提供了人员、制度的保障。

4．贴近应用软件的测试工作实际，寻找功能测试工具实施的切入点

在实施功能测试工具时，实施团队往往横冲直撞，如对试点项目未经深入分析是否适于应用现有的功能测试工具，是否在项目每个测试阶段均需要运用功能测试工具，即立即要求测试人员编制自动化测试脚本，立刻运行自动化脚本。这样实施的后果是：测试人员普遍感觉功能测试工具并没有给现有的测试工作带来多大的改善，反而是一种额外的负担。工具的实施者和具体使用者的矛盾逐渐突出，这也是功能测试工具失败的导火索。因此实施功能测试工具应找到一个比较贴近应用软件测试工作实际的切入点。而功能测试工具实施后能立竿见影的给测试人员带来帮助的实施策略是最佳实施策略。

功能测试工具最大优点是可不断重用自动化测试脚本，若将功能测试工具实施初期阶段定位在带有强烈重复工作性质的回归测试上，则测试工具的使用人员会立刻感受到功能测试工具所带来的明显优势，会抛弃初期排斥心理而对工具产生好感。特别在测试过程中过程版本构建频繁、回归测试工作量较大的情况下，其实施效果更加明显。因此建议实施时将回归测试作为功能测试工具实施的最初切入点。

5．建立手工案例、自动化脚本的版本管理机制，并与应用程序的版本管理对应

在功能测试工具实施一段时间之后，实施者常常发现管理、维护已经建立的脚本需要耗费大量的工作，一旦脚本管理失控，后期维护未能及时跟进，则花费大量心血建立的自动化脚本将会付之东流。因此如何管理好已经建立的脚本是功能测试工具实施时必须要做的一件事情。

首先手工案例必须与软件需求建立关联，也即与具体的业务功能建立关联；同时自动化脚本必须与这些手工案例建立关联。可以通过自动化脚本的“配置管理工具”对自动化脚本进行权限管理和版本管理；且整个自动化脚本的版本管理，必须与应用程序的版本管理建立关联，也就是当应用程序发生变化时，测试人员可以知道哪些自动化脚本需要及时维护。例如目前较多的软件企业已经布局Firefly（Hansky公司）、CVS（开源）、ClearCase（IBM）等工具进行代码、文档的配置管理，我们只需要将这些配置管理工具向测试领域延伸，利用这些工具对手工案例、自动化测试脚本实施配置管理。这样既不用花费额外的工具成本，又可以很好地实现脚本的版本管理以及与应用程序代码的联动管理。

6．设计便捷的数据维护策略

功能测试工具实施者需要在数据的可维护性和脚本的独立性之间做出一个权衡，需要根据本软件企业自身的特点，来决定采取何种自动化脚本的数据维护策略。

举例来看：如果开发企业的产品（如证券、大型银行、电信的产品）多在大型计算机上运行、或者多个复杂的平台环境（如跨机构的平台）上运行，其测试环境的备份、恢复耗费的时间较长，故建议其采取的数据维护策略是：建立脚本与数据的紧密联系，保证脚本运行的相对独立性，以适应庞大而无法灵活恢复的测试环境。反之如果是开放平台的应用，则可以采取脚本与数据分离的策略；若开发企业多是业务测试人员（较少的技术背景）从事功能测试工作，则应该采取数据与脚本分离的策略，以保证脚本的易维护性。反之若功能测试人员多是具备开发背景的人员，可以采取数据与脚本稍紧密的维护策略，保证自动化脚本的健壮性。

结语

软件测试领域是一个比较新的领域，国内一些应用软件开发企业越来越重视应用软件的测试工作，功能测试工具也逐渐进入到应用软件开发软件企业当中。此文仅简单探讨一下目前应用软件开发企业实施功能测试工具遇到的常见问题，希望能达到抛砖引玉的目的，引发业内测试专家对功能测试自动化工作展开讨论。■

作者简介

吴海荣，中国工商银行股份有限公司软件开发中心测试高级经理。目前在中国工商银行股份有限公司软件开发中心任职，从事金融软件的自动化测试研究和项目管理工作。

■ 责任编辑：郑柯 (zhengke@csdn.net)

CUDA 编程之线程执行

或许看到下面的内容的时候，你会觉得和传统的讲解线程，和一些讲解计算机的书的内容不是很相同。我倒觉得有关计算机，编程这些方面的内容，并不都是深奥难懂的，再深奥难懂的事情，其实本质上也是很简单的。一直以为计算机编程就像小时候搭建积木一样，只要知道游戏规则，怎么玩就看自己的。或许是从小学那会，就喜欢在做数学题的时候用一些简便方法来解题，养成了一些习惯，喜欢在遇到复杂问题的时候尝试用最简单的方法来解决，而不喜欢把简单的问题弄得很复杂。

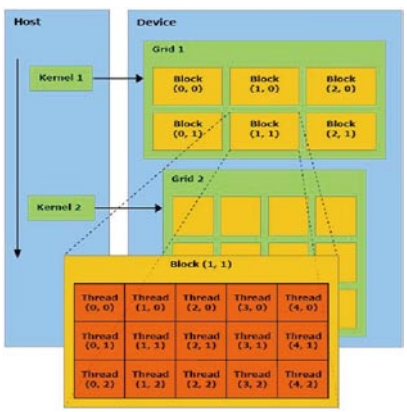
要真正进入CUDA，就必须先了解CUDA线程运行模型，才能在这个基础上做并行程序的开发。

CUDA在执行的时候是让host调度一个一个的kernel按照线程网格（Grid）的概念在显卡硬件（GPU）上执行。每一个线程网格又可以包含多个线程块（block），每一个线程块中又可以包含多个线程（thread）。

在这里我们可以拿古时候的军队作为一个例子来理解这里的程序执行模型。每一个线程，就相当于我们的每一个士兵。当要执行某一个军事任务的时候，大将军把任务分成M个部分，每一个部分做不同的工作，有的是做侦查的工作，有的是做诱敌的工作，有的是做伏击的工作，有的是做后备的工作，有的是做后勤的工作……反正把一个大任务按照不同的类别，不同的流程不同，分别由M个部分来完成。

这里我们可以把大将军看着是Host，它把这次军事行动分解成一个一个的kernel：kernel_1，kernel_2……kernel_M，每一个kernel就交给每一个Grid（副将）来完成。当要执行这些任务的时候，每一个Grid又把任务分成工作相同的子任务。由一个个的block（百户）来进行管理，这里每一个Grid管理的Block也是有限的。每个block(百户)就管理相应各自的thread(士兵)。

由于古代通信不是很方便（从GPU的发展史来看，如果按照中国的历史，现在的GPU也就还处在战国时代），所以每一个Block（百户）内部的Thread（士兵）才能方便的通信，按照既定的规则进行同步；而各个block之间就没那么方便了，大家不能互相通讯。不过同一个（千户）Grid管理的block之间是共享同一个任务分配的资源的。每一个Grid都可以从大将军那里分配到一些任务，和一些粮食，同一个Grid的block都可以分到这个Grid分配到的粮食。而每一个（千户）Grid本身的任务就不一样，所以Grid除了知道自己做的事情外，其他的Grid他都不会知道了。下面让我们来看看在GPU中东图例说明：



看到这张图，我们可以对应来讲解我们的Thread部队。一个大将军Host，分配了任务中的两个任务（Kernel1，Kernel2）给了千户（Grid1，Grid2）来完成。千户Grid1里面把自己的队伍分成了6个百户Block，然后每一个百户又把任务分配给了自己的士兵（Thread）来具体完成。这里得说明的是，由于千户拿到的任务Kernel是定了的，所以到每个士兵（Thread）也就那里就只会埋头做同样的事情（就像戚继光招的兵：在胡宗宪的幕僚郑若曾所著的《江南经略》中，有着这样一份详细的招生简章，如果不服气，大可以去对照一下：凡选入军中之人，以下几等人不可用，在市井

里混过的人不能用，喜欢花拳绣腿的人不能用，年纪过四十的人不能用，在政府机关干过的人不能用。以上尚在其次，更神奇的要求还在下面：喜欢吹牛、高谈阔论的人不能用，胆子小的人不能用，长得白的人不能用，为保证队伍的心理健康，性格偏激（偏见执拗）的人也不能用。……概括起来，戚继光要找的是这样一群人：四肢发达，头脑简单，为人老实，遵纪守法服从政府，敢打硬仗，敢冲锋不怕死，具备二愣子性格的肌肉男。——《明朝那些事儿》)。

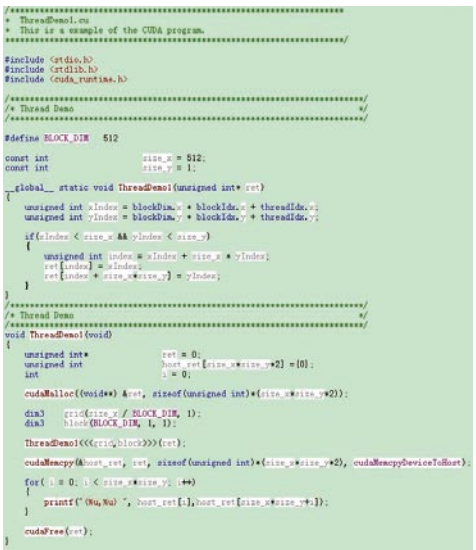
为了方便统一管理，大家都去掉了自己的名字，按照Grid1，Block（x，y），Thread（x，y）这样的编号来称呼每一个Thread士兵。如果你要找到某一个Thread，你就跑到军营里面大叫：喂，Grid1手下的Block1管理的三排第二个Thread（1，2）出来。对于每个士兵自己来说，他要知道自己的位置，就得知道自己的长官都是谁。Thread（1，2）要知道自己再整个Grid手下算第几个兵（钢七年第……个兵），当Grid1叫到他的号了，他得马上回答：我在Block（1，1）的编号是：

```
unsigned int xIndex = blockDim.x
* blockIdx.x + threadIdx.x;
      unsigned int yIndex
= blockDim.y * blockIdx.y +
threadIdx.y;
```

如果要得到线性的编号，我们可以自己算一下：这个士兵是在Grid部队下的第xIndex行，yIndex列站着如果要得到线性的编号，我们可以自己算一下：这个士兵是在Grid部队下的第xIndex行，yIndex列站着（这里我们必须注意：blockDim.x（这里为5），blockDim.y（这里为3）不是block的坐标，这里是block的size，切记！）。如果从第一个士兵哪里算是线性编号0，那他的线性编号就是

unsigned int index = xIndex（6）+ size_x * yIndex（5）；这里的size_x就是一行一共有多少个士兵（Thread），例如上图，这里一行有3个block每一个block里面的每一行有5个Thread，所以size_x就应该为3×5=15，一个Grid的一行有15个士兵，那刚才叫道的那个人的线性编号就应该是……还要我算吗？如果不得81，自己再算一下……（编号是从0，开始的）计算过程：index = 6+5×15=81；

讲了这么多，咱也不能光说不练假把式。下面给一个简单的Thread 测试的Demo。本来打算把整个代码都copy过来，但是考虑到又会被别人copy出去，这样copy来copy去，在编程中很容易出现错误，所以作者也不提倡在做编程中直接copy代码，这样很危险的，很多自己都不知道的bug就隐藏在copy的代码当中，下面截图：



作者这里做了一个简单的测试，测试了512个线程。这里只有一个grid，从这一个grid里面也只分了一个bock：

```
dim3 grid(size_x / BLOCK_DIM, 1)
---> dim3 grid(1, 1);一个Grid里面一
个blcok
dim3 block(BLOCK_DIM, 1, 1)
---> dim3 block(512, 1, 1);一个
Block里面分配512个Thread;
```

这里的每一个任务kernel就是：

```
__global__ static void
ThreadDemo1(unsigned int* ret)

{
unsigned int xIndex
= blockDim.x * blockIdx.x +
threadIdx.x;
unsigned int yIndex
= blockDim.y * blockIdx.y +
threadIdx.y;
if(xIndex < size_x &&
yIndex < size_y)
{
unsigned int index
 = xIndex + size_x * yIndex;
ret[index]
 = xIndex;
ret[index + size_x*size_y]
 = yIndex;
}
```

计算自己的线性id然后把自己的坐标写入到线性id对应的数组里面。Ps：说明一下，这个记录id坐标的数组ret[],ret的前一半记录的是线程的x坐标，后一般是记录的y坐标。（CUDA是C的扩展，这里的const定义的常量的用法在ANSIC C里面是行不通的，但是在C++中是可用的。）

每个任务kernel都说好了，然后就是host下达命令：

```
      ThreadDemo1<<<grid,block>>>
(ret);
```

来运行程序。所有的士兵都开始工作了，把自己的坐标x，y写入到ret数组里面。

下面是得到的结果：

(0,0) (1,0) (2,0) (3,0) (4,0) (5,0) (6,0) (7,0) (8,0) (9,0) (10,0) (11,0) (12,0) (13,0) (14,0) (15,0)…… (510,0) (511,0)

一目了然，正是我们定义的512个线程。

前面我们通过一个block定义了512个线程来演示了线程的运行过程。在后续文章中，我们会用士兵用餐的例子来讲解多个block在同一个Grid中运行的模型。■

用户为中心设计（下）

上期探讨了用户为中心的设计准则。这一期，我们来看看有哪些典型现实技术问题。

■ 文 / 王翔

亲和力

正如我们相对更倾向于与具有亲和力的同行共事一样，用户也更倾向于使用“亲和力”好的应用或服务。其中用户为中心设计的内容主包括以下几点。

1. 提供精简的信息获取方式和操作指示

不可否认，在很多应用中，用户不一定必须严格按照谁先谁后的操作步骤。例如：注册新用户的时候，用户愿意先填Email还是先填他感兴趣的板块其实无所谓，但如果40多项注册内容全部“平摊”在用户面前，每次点击“Accept & Submit”的时候被及时反馈了一大堆不合规内容时，相信几个来回之后，在这个人和网页的PK中，人的心理就处于下风。可能会有某些用户就是喜欢这种“山重水复疑无路，柳暗花明又一村”的注册页面。这40多项注册内容，即便先后无关，我们也可以设计为按照先后不同Step填写项目，这样一来，用户就可以按照即定的过程填写当前功能需要的信息。即便存在错误也仅仅限于个别内容项，用户也不需要记忆那么多“A部分A1、A2、A3错误”、“B部分，B1、B2…错误”，效果可能反而更佳。

用户使用信息系统，毕竟绝大多数情况下还是希望简单、省事。虽说水立方到前门换乘路线很多，但用户之所以咨询信息系统，他的目的还是希望了解最便捷的那一条。其他系统也是，太多的选择反而容易让用户没法选择。

2. 将信息分块

上面是对操作的划分，对于信息内容也是。我们了解信息时往往是左右脑分工记忆，但如果长篇全都是文字，那么左右脑分工失衡，一方面不利于用户对于信息的获取，另外信息也在系统付出了更多的处理后也“费力不讨好”。用户为中心的信息分块并不等同于我们常看见的“下一页”，他强调在把信息分解之后，最好还以人易于认知的方式组织信息的架构。例如下面的文字：

“上大学时，妈妈在车站送我的情境依然历历在目，每每时常回忆起来。”

从文学角度看，这样的信息比较充实，但对于使用信息系统的用户而言，如果每条股票涨跌信息都附加各种修辞和感情色彩，就会变得有点类似电视促销——“Yeah, 这支股票又以如虹的气势上涨了3%”，类似做法既不利于用户的访问，同时也违反了“减少用户记忆负荷”的原则。而用户为中心设计则更倾向于应用系统将信息组织成图1的方式，然后再呈现给用户。

图1只是一种实现方式，用户为中心设计对于具体实现技术并没有要求，实际项目中我们可以用XML、思维导图、动态文本、Tag等方式将分割为块的信息组织在界面上。这样用户只关注于他希望了解的内容以及需要掌握的细致程度即可。

“信息分块”组织及下面的“提供简要介绍”其实也在告诫我们要扭转应用设计的思维，不是“我让用户看什么”，而是以“用户要看什么”、“用户要怎么看”为设计的前提。

3. 提供简要介绍

作为“信息分块”的一个补充，用户为中心设计强调可能的时候，要对信息增加必要的线索，让用户可以根据线索中呈现的只言片语快速定位到它需要的内容上，然后获取相关信息块的内容。以往我们做开发的时候，总喜欢抱着这样的心理——“他自己打开不会看啊”。事实上，除非软件仅显示只言片语或者信息内容简单到“有能耐你整四岁的，五岁都不在乎~~~”，否则建议对信息增加“扼要”部分，用户根据“扼要”决定这是不是自己需要的内容，判断是否有进一步阅读的必要。

这条其实在一些经常信息过剩的应用中贯彻的很好，例如Google Reader和图2中的Outlook。

其他

除了上面3条外，用户为中心设计还有几个主要的设计准则。

1. 定位和导航

- 用户应当能够可视化了解他所

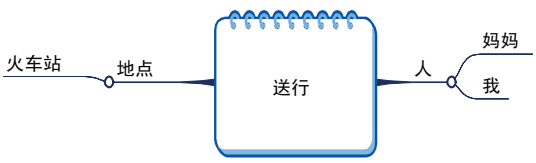


图1 思维导图方式表示的信息层次关系

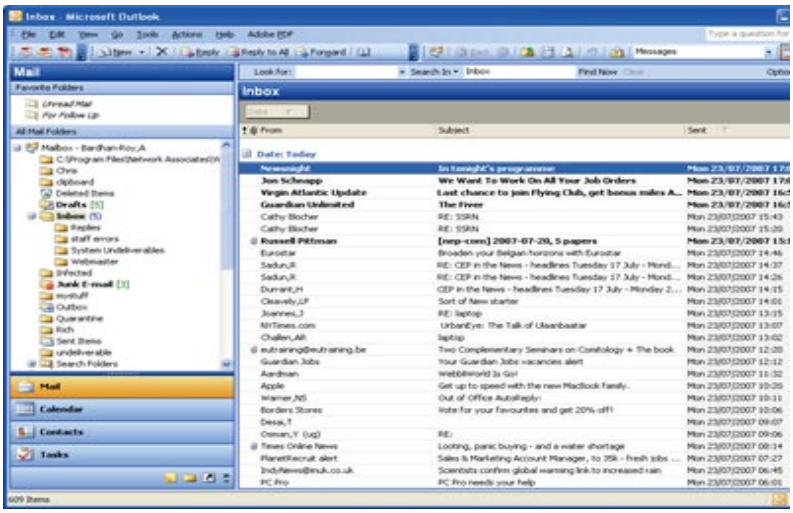


图2 带有扼要信息的Outlook

能操作的功能。

● 应用不应该有“死胡同”，最起码用户要知道如何退出当前功能。

● 界面要节制用户的操作，最好仅提供当前上下文环境下（甚至是多信息源、服务源环境共同约束下）只能执行的操作。

2. 错误

如果可以贯彻上述原则，那么用户界面上无关元素的数量最小化，出现错误链接、错误导航的情况也预期会降低。尽管如此，应用还是会面对一些错误，用户为中心设计的建议是尽量减少用户错误的绊脚石，如果出现错误要尽可能提供迅速恢复（哪怕是部分恢复）错误的机制。

这一条对消费性应用而言，还显得不那么重要，但对于电子商务应用，尤其是涉及电子货币的应用就显得尤为关键。试想，当用户提交一个“大事务”——下订单，结果浏览器的小圈转了几圈之后出现一个不冷不热的错误——超时，虽然没有显示“提交失败”，但页面其他内容基本正常刷新后也没有显示“提交成功”，这时候您的用户难免会补充几句负面评价。

3. 语言

● 尽量不要出现批评用户的语言，例如：“您提交了一个非法的Email”、“您的操作非法”等等，毕竟我们的应用面对Leo那种高级黑客的机会估计比一天中2次头彩的几率可能都小，用户看到这些错误绝大部分是因为操作疏忽或者对应用不了解，经常出现这类惩戒性用语会大大降低用户使用过程中的“愉悦感”。

● 使用短句。

● 尽量使用日常用语。

● 即便要体现幽默也要考虑到您的用户，尤其在一些国际化软件中，隐喻、图标、成语、双关语都要慎重。

4. 文字可读性

● 虽然采用某些装饰字会让您的应用看上去很Cool或者“粉可爱”，但装饰字体难以阅读，除非万不得已，否则不要使用。

● 斜体字也不适于阅读。

● 正文太大或太小也难以阅读。

● 具有引用和链接的内容要尽量明确提示给用户，而不是

图3 Google Translate的界面

当用户鼠标滑过的时候才出现一个下滑线或一只可爱的小手。此外，链接前最好提供部分“扼要”介绍。

如图3所示：Google Translate为了便于用户阅读，减少用户记忆负载，在显示译文的同时还会提示原文内容。

- 对于滚动信息，如果文本块长于50字符要大大放慢（2倍~5倍）滚动速度。
- 也许朦胧是您表达美感的常用手法，但文字配色如果朦胧了您的用户就要吃点苦头了。

用户为中心设计中典型现实技术问题

上文我们列出了很多准则，除此以外，用户为中心应用的开发过程与普通软件工程过程区别不大，不过强调开发过程中更多站在用户角度思考而已。但技术实现上，我们必须面对几个问题。

- “应用找用户，而非用户找应用”。虽然说起来容易，但在网络、应用林林总总的企业环境中确有很多障碍，而且多数都是历史人为造成的。
- 企业往往将核心网、普通办公网分割，但站在用户角度，这些本不是他们应该关心的问题，用户为中心设计需要解决跨网问题或者必须依赖企业统一的跨网交换机制。
- 由于用户账号管理参差不齐，同一用户在不同系统中往往登记为不同的账号，有些用户又经常在同一系统内拥有多个账号。这样问题就来了，应用找谁？应用A认识Ua1, Ua2, 应用B认识Ub1, Ub2, Ub3，但其实Ua1和Ub2就是一个人，因此用户为中心设计中必要完成这个账号的映射及登记机制，或则凭借ESB或Cloud的Unified Identity & Authentication Service的统一用户管理，以桥方式完成应用找用户的过程。
- 不同应用的授权机制、任务生成规则均有差异，除了要用户统一管理外，还要把相关安全机制施加到用户为中心设计中。

表1 典型的服务元语设计

项　目	指　标	说　明
服务元语	新增 提取 撤销 置换 追加 重发 挂起 恢复 反馈 测试（用户监控和开发、调试）	在应用找用户的过程中，以服务元语的方式描述每个任务消息的性质，便于任务进行修改和补充

- 不同应用采用的开发技术也是林林总总，客户端环境安装的操作系统、数据访问组件、.NET Framework、JDK版本（包括其他语言的Runtime）也不同，需要提供兼容性设计，对于实施了SOA的企业，基于XML标准及WS-*协议的应用比较好组织他们集中找用户，而没有实施SOA的企业，如果开发技术又比较杂，实施难度会很大。
- 用户为中心设计上除了支持正向操作外，还需要考虑到“应用找用户”时，可能引起的“反操作”。例如：撤回、取消相关任务，这对于用户为中心的缓存和持久化设计都提出更高的要求。例如：可能要支持表1中的服务元语。

相关内容

除上面介绍的内容外，用户为中心设计还对于可用性测试有额外要求：可用性测试从需求评审开始贯穿整个开发周期，对于一些可用性要求高的项目，甚至可用性分析可以反超到可行性研究及需求获取阶段。

例如：用户看了《少数派报告》后，然后坚持下一个OA系统“就照那个设计”了，面对这种情况，您恐怕还是先做个可用性、可行性分析比较稳妥。

用户为中心的设计涉及认知心理学、人类学、人机交互、视觉和图形艺术、通讯、语言学、色彩理论、印刷术等很多内容，但很少有开发团队可以请得起这么多人员专门作可用性设计，这时不妨参考业内大公司的一些CheckList。例如，表2就是是裁减后的某公司一份80年代的样本（那时候还暂时不用考虑跨应用的服务访问）。

表2 一份较早前典型的检查单

项　目
1. 我觉得我会常常使用这个系统
2. 我发现这个系统人为作得非常复杂
3. 我希望这个系统操作更简便
4. 我觉得使用这个系统需要有IT人员协助
5. 我觉得功能组织不好，和机构职能有冲突
6. 系统中有些地方功能设置得很矛盾
7. 我觉得大多数人经过培训都会很快上手
8. 我觉得这个系统运转得很慢
9. 找到我常用的功能比较费力，经常需要求助IT
10. 我有信心使用这个系统

从表2的检查清单几乎看不到任何专用技术术语，但相信凭经验您会发现越是这样的内容越不容易实现，现在很多需求分析师总是通过“诱导”的方式让用户接受一个个折衷的方案，确实实现用户为中心设计需要甲方、乙方付出更多代价，需求、开发、测试周期也要相应延长，相信这也是很多项目和产品最后还是走“机器为中心”路线的原因。但如果开发中注意上面提及的一些准则，我们起码可以用较小的代价换取用户更好的评价。

当然，用户为中心设计也要提供“两厢情愿”的机制，否则很容易与垃圾短信的感觉画等号——很烦很无奈。

用户为中心设计除了是个方法和过程外，应该也是软件行业一个美丽的目标和愿景。■

作者简介

王翔，软件架构师，主要研究方向为XML、.NET、领域设计和PKI应用。工作之余喜爱旅游、写作和烹饪。

■ 责任编辑：郑柯 (zhengke@csdn.net)

Scala上的Twitter

Twitter将部分应用从Ruby迁移到了Scala。三位开发者详谈决策背后的因素、Scala在现实应用中遇到的困难以及Scala对编程风格的影响。

■ 文 / Bill Venners

Twitter是一个快速成长的微博客服务网站。它作为一个Ruby on Rails应用呱呱落地，现在依然用Ruby on Rails支撑着大部分面向用户的网页。但大约一年以前，他们开始将部分Ruby服务替换成用Scala编写、在JVM上运行的程序。来自Twitter的三位开发者——系统工程师Steve Jenson、API主管Alex Payne、服务团队成员Robey Pointer——与Bill Venners坐到一起讨论Scala在Twitter的实际应用。他们详述了运营中遇到的实际问题如何使他们开始考虑Scala，Scala在实践中又产生哪些问题，还会探讨Scala怎样影响了他们的编程风格。

Twitter简短回顾

Bill Venners：在Twitter的技术演变历程里，是什么因素令你开始考虑Scala？

Alex Payne：Twitter最初只是从事播客业务的ODEO公司里面私下鼓捣的项目。当ODEO遇到经营困难，他们开始允许工程师折腾各种想法。其中一位工程师Jack Dorsey对状态尤为感兴趣。他总是在IM上看到有人说“我在遛狗”，“我在做这个”，“我在干那个”，于是开始思考有什么办法能让人们更方便地交流各自的状态。他和另外几位工程师搭建的原型后来成了Twitter。原型沿用了ODEO惯用的Ruby on Rails。直到今天Twitter都主要是一个Rails应用加若干Ruby守护进程在后台完成异步处理。

后来我们发现虽然Rails做前台Web开发非常出色，但对重量级的后台处理来说，Rails在运行时有一些性能上的局限。我个人认为Ruby语言缺少一些有助于产生可靠的高性能代码的因素，而当我们的业务越做越大，这样的因素对我们来说非常重要。我们希望写出来的代码是正确的、可维护的。我们还希望维持低成本——和大多数企业对软件平台的期望没什么两样。所以我们开始盯上Scala。

我们看中Scala还有一个原因。虽然我们用Ruby遇到了问题，但依旧热爱Ruby语言的灵活性、喜欢它功能全面、也忘不了用Ruby编程的愉悦 。很多Java开发者离开大公司之后都转而用Ruby编程，原因和我们一样，希望每天都过得愉快。我们不希望丢掉这些，投向某种无趣的、一本正经的语言社群，比如C++。我们知道人们用C++写出非常高性能的代码，在座的Steve和Robey都有那方面的经验。可是我们希望能用一种真正令我们倾心的语言，而Scala令我们感觉值得赌一把。

可靠的高性能代码

Bill Venners：我很好奇，Ruby粉丝也会希望你说清楚——能否详细解释对于编写可靠的高性能代码来说，你感觉Ruby缺少了什么？

Steve Jenson：在我的职业生涯中，一直觉得长生命期的进程是不能缺少的东西。而Ruby，像很多脚本语言一样，作为长生命期进程的环境会遇到麻烦。相反JVM非常适合，因为过去十年JVM都一直在为这个方面优化。所以Scala具备了编写长生命期服务的基础，而这也是目前它在Twitter的主要用途。Scala还有一样特质令我们爱不释手，就是静态类型，而且是无痛的静态类型。有时候在Ruby里面会很希望能写一些可选的类型标注，特别指明在某个地方我们希望出现什么类型。Scala就能指定这样的类型信息，我们感觉特别有用。

Robey Pointer：另外Ruby还没有很好的线程支持。现在情况改善了，不过当初我们编写这些服务的时候，绿色线程是唯一的选择。绿色线程不用真正的操作系统内核线程，它们定期停下手上的工作，检查一下有没有别的“线程”要执行，用这样的办法去模拟。所以Ruby是在单核或者单处理器的范围内模拟线程。我们用多核服务器，而且服务器上的内存是有限的。在欠缺很好的线程支持的时候，就只能依靠多进程了。但因为Ruby的垃圾收集不像Java那么完善，每个进程都要用掉很多内存。没有足够大的内存我们就没办法在单台机器上跑很多Ruby守护进程。当我们用JVM的时候，可以在堆里跑很多线程，还可以让单个JVM进程占据机器上的全部内存。

Alex Payne：我要特别强调Steve刚才提到的“类

型”。随着我们的系统越来越大，我们的Ruby代码里面有很多逻辑可以说是在模仿一个类型系统，单元测试或者模型验证的时候都容易出现那样的逻辑。我想这种情况可能是动态语言编写的大型系统与生俱来的——你最终不得不自己重写一个类型系统，而且很可能写不好。到处都要检查null值，到处都调用Ruby的kind_of?方法。“kind_of?”相当于问“这个是User对象吗？不好意思，我们一定要那种的，不然程序会爆掉。”写这样的东西实在是太丢人了，在编程语言的世界里，分明几十年前就已经有了解决方案。

用Scala补足Ruby

Steve Jenson：我们发现Ruby和Scala非常互补。我们把Ruby，具体说是Rails，用在它特别擅长的地方。前端的东西Rails做得很好。

Bill Venners：你们把Scala用在什么地方？

Robey Pointer：我们有一个基于Ruby的队列系统负责Rails前端和守护进程之间的通信，现在换成了用Scala重写的队列。在情况稳定的时候，原来的Ruby队列其实表现得还不错，只不过启动时间和崩溃行为不符合我们的期望。它慢了一点，内存消耗多了一点。有时候遇到流量高峰就撑不住了，而当它崩溃之后，要很长时间才能恢复。这可不行，我们需要能对付极端情况和高负载的方案，即使不像对付一般负载那么轻松，至少要相对容易。

Bill Venners：守护进程做些什么？

Robey Pointer：我们的架构有一个重要的出发点，就是让Rails做它所擅长的事情，AJAX、Web前端、网站——总之就是用户看得见的东西。任何能脱离请求/响应循环的工作，我们都会把它卸下来，放进消息系统里面排队，让后端的守护进程去处理。

Steve Jenson：例如当你修改社会关系图的时候，也就是你在Twitter上“跟”或者“不跟”某个人的时候，这些工作以及相应的缓存更新都由守护进程异步完成。

Bill Venners：你们考虑过JRuby吗？

Alex Payne：考虑过。只不过当时试验的时候，没办法在JRuby上启动我们的Rails应用。我们用了大量要求C扩展的Ruby Gems，而当中很多都还没移植成JVM友好的版本。当时JRuby的性能还远不能与MRI（Ruby的C实现）相提并论，更没法与Scala这样的语言相比了。我们不排除将来会再次尝试JRuby，但目前只是希望给Ruby打些补丁能有帮助。

Scala的权衡取舍

Bill Venners：你们已经有了Scala的实战经验，用它去解决实际的问题。你们认为它有什么代价，有什么问题？好处在哪里，坏处在哪里？

Steve Jenson：我认为Scala的效果很好。我们当中很多人都有过使用研究型语言编程的经验，而一般来说，当把研究型语言投入实际生产会遇到很多问题。可是我们用Scala的时候没有遇到太多这类困难。我知道在Actor和高可伸缩性方面遇到一点麻烦，不过已经解决了。总体来说，从我们的经验看，Scala是一个非常高性能、非常稳定的系统。

Robey Pointer：我也同意到现在为止遇到的问题非常少。其中一些是由于语言和编译器都还很新。我们偶尔遇到一些令人困惑的编译错误，要花一些时间才能找出真正的错误。Scala有一部分核心集合库还不及格，当然现在他们正在改进。

Bill Venners：哪方面“不及格”？用不了？不够快？

Robey Pointer：我没有遇到用不了的情况，但有那么几个方法写得效率不高，API也存在一些缺口。有的时候我们干脆决定丢开它，直接在Scala里面用Java的集合库。这倒是Scala的一项优点，它有Java做后备。

Alex Payne：我首先做的一件事是给我们的API准备Scala测试平台。主要是包装Apache Commons HTTP库，还要提供代表系统中所有RESTful资源的一组对象。最困难的地方是把Ruby思维换成Scala思维。要从更加函数式的角度去思考，还要更加从不变性的角度去思考，更何况我好几年没用过静态类型的思考方式了。所以对于Java背景不那么强，更偏向动态语言背景的人来说，过渡期可能长一点，不过只要过了那个阶段，结果是值得的。现在我的默认思维已经用Scala取代了以前的Ruby。

Bill Venners：学习Scala如何改变你对编程的想法？

Robey Pointer：学习Scala之前，我没有比Python更强的函数式语言背景。我很熟悉Python。越深入学习Scala，我就越从函数式的角度去思考。一开始的时候，我会像写Python一样用for表达式，现在我发现自己经常会用map或者直接在迭代子上调用foreach。

Alex Payne：我感觉从Actor的角度去思考并发绝对是一次思维转变。我有过一点用IO语言编程的经历，不过我感觉Scala的Actor更接近Erlang的模型，与IO差别更大一些，我非常喜欢Scala的实现。这是好的转变。

Steve Jenson：我来自Java背景，不过我也熟悉Common LISP和ML。能用自己熟悉的运行时，又能用函数式的组合子、闭包和高阶函数，这是美妙的事情。它们在Scala里面的表现我很满意。

Scala的顾虑

Bill Venners：如果我打算在生产系统中使用Scala，有什么需要担心的呢？哪些事情是我必须先准备好的？有什么要害怕的吗？

Alex Payne：目前Scala的IDE支持情况，应该说已经过了婴儿期，但还处在别扭的青春期。IntelliJ 8.1的

Scala支持感觉相当不错。Scala自带的Emacs模式，缩进有些怪毛病。Textmate的支持很差劲，不过Scala工具邮件列表上已经在讨论怎么改进。IDE支持算是个门槛。

Robey Pointer：如果你不是从Java世界过来的，而是来自Ruby或者Python的世界，有可能受不了编译-部署的流程。你要配置构建环境，还要用很长的脚本去部署jar文件，与Ruby或者Python相比，这个世界差别很大。

Alex Payne：JavaRebel对此有些帮助，把它配置好之后就比较接近以前的流程——写代码、保存、运行测试。情况有所改善，但还是免不了Java世界的拖泥带水，每个项目都要先写一堆配置。不过会得到一套好的惯例，增加新的库很方便，还有很多部署相关的东西都已经内建在里面了。终究是个权衡问题。

Steve Jenson：要保证把可变性（mutability）用对了地方。先从不可变开始，当发现适合的时候再运用可变性。这就是我们得到的教训。应当注重不可变，理由是当用到线程的时候，如果对象是不可变的，就无须担心发生意外的变化。这个原则对我们来说是一大收获，现在如果不是特别需要提高性能的地方，都尽可能不要可变性。

Robey Pointer：JIT编译器还能对不可变对象做一些很重要的优化，大大提高性能。

Alex Payne：我们还遇到一种情况，肯定属于很特殊的情况，大家不要因此被吓倒了。我们的Hosebird服务是一个独立的专用系统，它将全部的公开Tweet消息流通过互联网近乎实时地发送给各家合作伙伴。最初系统里有很多Actor：一个负责从内部的消息队列上抓消息，还有很多个Actor各自代表不同的客户。随着我们做的系统测试越来越多，我们发现Actor不一定是适合系统所有部分的最佳并发模型。现在系统里一部分保留了基于Actor的并发模型，其余部分干脆回到了传统的Java线程模型。负责的工程师John Kalucki认为这样更容易测试，更好预测。最妙的是只要几分钟就能把基于Actor的代码换成基于Java线程的代码，几次查找替换而已。所以万一发现Actor不适用了，也不是什么大不了的事情。

Robey Pointer：我在队列系统Kestrel也遇到同样的情况。一开头是每个队列一个Actor，但发现任务的粒度太细了，在那么细微的层次上用Java的锁实际效果更好。

Bill Venners：*在现实世界中使用Scala还有什么地方要注意的吗？*

Alex Payne：如果一个程序员从来没有用过支持模式匹配的语言，那要准备好改变自己对编程的认知。我和一群用Objective-C的Mac程序员谈过，试图说服他们，一旦你开始用模式匹配，就再也不想回到其他语言了。模式匹配是程序员每天都在做的事情。给我一个集合，我让从大海里面找出特定的那一根针来，根据类去找也好，根据内容去找也好，模式匹配就是那么强大的一个工具。

Robey Pointer：我还想说说我们是怎么开始用Scala的。绝不是哪天晚上喝多了两杯拍脑袋做的决定，而是挣扎了很长时间，也许算不上挣扎吧，但至少讨论了很长时间。当你用过像Ruby这么高级的语言编程，再回到中级的语言，比如Java，会感觉不耐烦，因为要多写很多代码才能达到同样的效果。Scala在这方面很吸引，因为可以继续写高层次的代码，只不过是在JVM上运行。

付诸行动

Bill Venners：*不久前JavaPosse的人说过他们是怎么尝试新事物的。他们建议应该在你关心的事情上尝试，但是不要在关键的业务上尝试。因为是你关心的事情，所以会一直保持进展；但要是在关键业务上尝试，一旦失败生意就完了。*

Steve Jenson：Twitter也是这么做的。我们首先做小规模的试验，用Scala实现“Public Timeline”服务。效果很好，我们也学到很多，知道了我们想要什么不想要什么。那个服务还在正常运行，已经运行了差不多一年。

Alex Payne：是的，唯一一次遇到问题是因为底层的数据库出现复制延迟。除此之外它都运行得很顺畅。因为太成功了，所以我们打算逐渐把更多架构迁移到Scala。我们的流量中API请求占大头，我们希望把其中大部分都交给Scala去处理，不管是Edge缓存层还是Web应用层。希望到2009年底的时候大多数用户与Twitter的交互背后都有Scala在辛勤劳动。■

参考资料

Programming Scala，Alex Payne与Dean Wampler合著：
http://oreilly.com/catalog/9780596157746/

Programming in Scala，Martin Odersky（Scala设计者）、Lex Spoon与Bill Venners合著：
http://www.artima.com/shop/programming_in_scala

Parleys.com上的视频The Feel of Scala：
http://tinyurl.com/dcfm4c

作者简介

Bill Venners，Artima Inc.总裁及网站的出版人。著有Inside the Java Virtual Machine一书。领导了Jini社区的ServiceUI项目，还主导面向Scala和Java开发者的开源测试工具ScalaTest项目的开发及设计。

■ 责任编辑：郭晓刚（guoxg@csdn.net）

Ruby并发之谜与多语Ruby

GIL限制了Ruby的多线程并发能力。多进程是一种答案，Ruby与Erlang并用的多语编程可能是另一种答案。

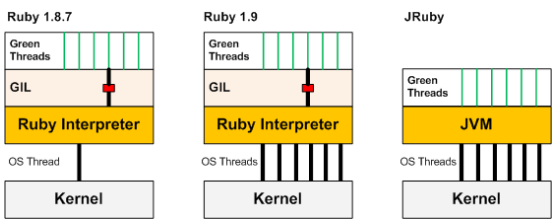
■ 文 / Ilya Grigorik

并发给我们的应用带来并行能力，而线程是实现并发的一种方式。可是在Ruby里面不是这么一回事：并行执行不等于多线程。如果你想让Ruby程序具备并行能力，应该向进程的方向去考虑。为什么呢？

为了弄清当中的秘密，我们要好好检查一下Ruby运行时。每当你启动一个Ruby应用的时候，就会有一个Ruby解释器的实例启动来解析代码、构建AST树，然后按照你的意思执行程序。这些对用户来说都是透明的。可是，作为运行时的一部分，解释器还会实例化一个全局解释器锁（GIL），而GIL正是削弱并发性的罪魁祸首。

全局解释器锁[1]是编程语言解释器线程持有的一种互斥锁，目的是为了避免与其他线程共享非线程安全的代码。每个解释器进程都有一个GIL。使用GIL严重限制了在单个解释器进程中通过多线程方式提高并发能力——在多处理器机器上执行该进程只能得到很少甚至没有速度提升。

解密GIL



为了说得更具体一些，让我们看看Ruby 1.8。首先给Ruby解释器分配了一个操作系统线程，实例化了一个GIL锁，接下来我们的程序再产生Ruby线程（绿色线程[2]）。你也看出来了，这个Ruby进程根本不可能在多核上得到什么好处：只有一个内核线程，也就是说在同一时刻只能执行一个Ruby线程。

Ruby 1.9看上去很有希望！Ruby1.9的解释器的确配备了许多原生线程，可是GIL却成了瓶颈。解释器要防范非线程安全的代码（包括你的代码和原生扩展），所以在同一时刻只允许单个线程执行。结果就是：Ruby MRI进程，以及任何采用了GIL的语言解释器进程（比如Python的线程模型与Ruby 1.9的模型非常类似），永远没办法享受到多核的好处！要想充分利用你的双核CPU，你只能运行两个进程。

实际上只有JRuby这个Ruby实现能让Ruby代码真正原生地运行在多核上。通过将Ruby编译成Java字节码放在JVM上运行，Ruby线程得以摆脱横亘在中间的GIL，直接映射到操作系统线程。这至少是考虑JRuby的一个很好的理由。

进程并行

GIL的影响之大令人吃惊，可是转念想想，这个难题的解法可能并不是那么复杂：与其琢磨线程，不如考虑一下如何在不同进程之间分配工作。这样一来不但绕过了并发编程中的一大类难题，还很有可能产生出一个具备水平伸缩能力的应用架构。我们需要做以下几个步骤：

1. 分解工作，或者分解你的应用。

2. 增加一个通信/工作队列（Starling、Beanstalk、RabbitMQ）。

3. 分叉，运行应用的多个实例。

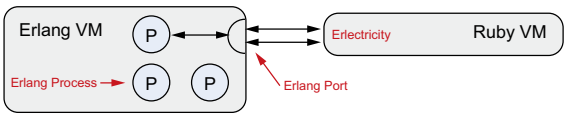
毫不出奇，许多Ruby应用已经采取了这样的策略：应用服务器集群（Mongrel、Ebb、Thin）已经是典型的Rails应用部署方案，其他逐渐流行的方案还有EventMachine和Revator（等价于Python的Twisted），无须在应用中引入线程，用很简单的方式就能实现并发的网络IO。

答案不一定只在Ruby身上

函数式语言，比如Erlang、Haskell、Clojure近一年半以来势头大有增长，分布式、容错、不停机的应用已经在眼前。其中最吸引人的可能是高并发的Actor模型[3]和消息传递模型[4]，尤其是考虑到GIL强加给Ruby程序的限制。也许这就是未来的方向？然而，当我跟随着Joe Armstrong

的入门杰作《Programming Erlang》步步深入的时候，不由得怀念起Ruby程序员唾手可得的众多第三方库。

Erlectricity架桥：连通Ruby和Erlang



好在Scott Fleckenstein和Tom Preston-Werner已经为我们解决了这个问题——通过Erlang端口[5]可以在Erlang和Ruby之间完成双向的数据交换。在Erlang看来，这种通信与Erlang进程之间的直接通信没什么两样，这就是说我们可以在任何需要的时候透明地调用Ruby解释器！准备好开始了吗，请执行：

gem install erlectricity

Erlectricity[6]使Ruby程序能够接收及响应通过Erlang二进制协议发送的Erlang消息。

来自Erlang的问候，由Ruby转达

Erlectricity源代码里包含了几个演示Ruby与Erlang接口的优秀范例：echo服务、用Gruff生成图表，甚至还有通过Campfire API聊天的例子，请到GitHub找Erlectricity代码库里的example目录。在这里，我们来看看如何连接Ruby和Erlang共同创建一则“Hello World”消息发到Twitter上。为了省事，我们借用一下John Nunemaker写的Twitter Gem[7]，不然用Erlang重新写一个Twitter API客户端太麻烦了：

```
> twitter-erl.rb
require 'rubygems'
require 'erlectricity'
require 'stringio'
require 'twitter'

receive do |f|
 f.when(:post, String) do |msg|

 # connect to twitter & post Hello World
 id = Twitter::Base.new('username', 'pass').
post(msg).id
 f.send!(:result, "Tweet sent! Status id: #{id}")

 f.receive_loop
 end
end
```

我们的Ruby进程可以经由Erlang调用，它会在接收状态监听并处理收到的消息。在这个例子里，我们仅仅模式匹配“post”请求字符串。接着看看Erlang那边的情况：

```
> twitter.erl
-module(twitter).
-export([hello_world/0]).

hello_world() ->
 Cmd = "ruby twitter-erl.rb",
 Port = open_port({spawn, Cmd}, [{packet, 4}, use_
stdio, exit_status, binary]),
 Payload = term_to_binary({post, <<"Hello World!
(from Erlang, through Ruby)">>}),
 % send command to Ruby script & wait for response
 port_command(Port, Payload),

 receive
 {Port, {data, Data}} ->
 {result, Text} = binary_to_term(Data),

 % print out resulting tweet
 io:format("~p~n", [Text])
 end.
```

Erlang代码稍微长一点，但依然非常简单直接。我们声明一个hello_world函数，打开一个面向字节的端口给Ruby解释器。通过该端口发送命令，然后阻塞起来等待响应。现在我们可以发出Twitter消息了：

```
Eshell V5.6.5 (abort with ^G)
1> c(twitter).
{ok,twitter}
2> twitter:hello_world().
<<"Tweet sent! Status id: 1358682092">>
```

多语未来：命令式+声明式编程

虽然这只是微不足道的一个小例子，但已经充分地显示出了多语言编程的威力。像Erlang这样的语言，或者说函数式语言、声明式语言，绝对会在未来占据一席之地，并且会扮演很重要的角色。但是，一种范式兴起并不意味着另一种就会衰落。事实上很有可能面向对象编程、函数式编程两者将结合在一起，共同支撑着未来的应用与架构。■

参考资料

[1]全局解释器锁：http://en.wikipedia.org/wiki/Global_Interpreter_Lock

[2]绿色线程：http://en.wikipedia.org/wiki/Green_threads

[3]Actor模型：http://en.wikipedia.org/wiki/Actor_model

[4]消息传递模型：http://en.wikipedia.org/wiki/Message_passing

[5]Erlang端口：http://erlang.org/doc/tutorial/c_portdriver.html

[6]Erlectricity：http://github.com/mojombo/erlectricity

[7]Twitter Gem：http://twitter.rubyforge.org/

作者简介

Ilya Grigorik，AideRSS创始人和CTO，博客作者、开发者。经常研究和鼓吹最新的Web技术、Web标准、云计算软件架构。热爱尝试一切数码新玩艺。

■ 责任编辑：郭晓刚（guoxg@csdn.net）

JVM不适合Erlang

Kevin Smith认为把Erlang移植到JVM是个坏主意。由于TCO递归、GC模型、序列化、闭包等方面的因素，对于一些具有强烈FP特征的语言来说，JVM平台是不明智的选择。

■ 文 / Kevin Smith

最近有很多人在讨论应该（必须）把Erlang移植到JVM，我恰恰认为这是个坏主意。对Erlang来说，JVM是个蹩脚的环境。尽管JVM是很多语言的好归宿，但我强烈相信只有BEAM VM（Erlang的虚拟机）才是最适合Erlang的环境，BEAM VM本身也是非常迷人的杰作。现在让我详细列举一下JVM不适合Erlang的理由吧。

JVM不支持TCO递归

TCO递归，也就是尾调用优化递归（tail call optimized recursion）是一种利用尾调用在常量空间内完成递归循环的方式。我举个例子，请看下面的伪代码：

```
foo(Counter)
  print Counter
  if Counter == 0
    return
  else
    foo(Counter - 1)
  end
```

在JVM上，程序可能会遇到栈空间不足的情况，因为每次调用foo()都要分配一个新的栈帧。在Erlang或者其他任何支持TCO递归的语言里，上述循环永远都不会分配超过一个栈帧。编译器（或解释器等）会检测到尾调用并重用已分配的栈帧。

TCO递归是Erlang仅有的循环方式。Erlang语言缺少其他常见的循环结构，比如for、do和while。所以缺少对TCO递归的原生支持，对于将Erlang移植到JVM的工作来说是一个极难克服的困难。

既然Erlang需要TCO支持，而JVM又没有提供，那么实现移植的唯一出路就是模拟。换言之，移植到JVM上的任何Erlang实现都要自行实现调用栈以提供TCO递归，而这一点将引发许多后果。其中最严重的影响是极大地损害了Erlang与Java的互操作。Java代码直接运行在JVM上，用的是JVM原生的调用机制和调用栈；Erlang代码则要运行在自己的栈实现之上。这两层之间怎么互操作呢？当TCO递归代码在一次递归循环中间调用Java代码会发生什么事？互操作一下子成了棘手的问题。解决此问题需要做多少工作，请参考SISC（一个基于Java的Scheme实现[1]）的情况。

JVM的单一垃圾收集器

任何达到相当规模的Erlang应用，都要运行成百上千个Erlang进程。这些进程运行的时候都在不断地分配数据、释放数据。到了一定时刻就必须执行垃圾收集清理那些被释放的数据，以免系统耗尽内存。问题是：怎样执行垃圾收集才能不影响系统的吞吐量和整体性能？

JVM提供了几种成熟的垃圾收集器，它们经过微调、优化，可以很好地应付JVM上常见的并发程度。可是对于移植的Erlang来说，问题出在JVM的收集器模型。整个JVM中只有一个收集器，而Erlang每个进程都有自己的垃圾收集器，目的是减少某个进程中发生的垃圾收集对其他进程的影响。我的看法是在Erlang程度的并发（数千个进程）条件下，JVM中的单一共享收集器比起Erlang每进程拥有各自的收集器，更容易成为瓶颈。

JVM对分布式编程的支持很弱

Erlang能优雅地扩展Actor模型，越过VM的边界，这种能力非常引人注目。只要稍加注意，本来在单个VM上运行的代码，可以完全不用修改就打散放到许多节点上运行。

JVM上根本找不到可以与此相提并论的东西。哦，通过第三方库可以做到稍微近似的效果，可是又怎及得上刻在骨子里的语言和运行时特性呢。外部库总是要做一些折中，配置、集成都是需要克服的干扰。你可能觉得如果把

这些库纳入Erlang4J运行时，就能做到对开发者透明，可是我认为在无可避免的折中代价与集成难题面前，最好的结局不过是一个半吊子的实现。

JVM对分布式数据没有可靠的支持

我可以用term_to_binary/1函数将任意Erlang term序列化成二进制形式；将得到的二进制序列通过网络送给另一个节点；在接收节点上对收到的二进制blob调用binary_to_term/1，将它变回Erlang。好吧，是有些例外。有很少一部分数据，比如文件描述符和套接字，对它们做序列化没什么意义。那我纠正一下，不是任意Erlang term，但我可以序列化绝大多数Erlang term。

JVM有Serializable接口，有ObjectInputStream、ObjectOutputStream以及一群作用相似的同伴。我认为它们都很失败。为什么？因为开发者从最初设计类的时候就必须将序列化纳入考虑，这些机制才有可能正常工作。而凭我的经验，绝大部分Java开发者都极少、乃至根本不考虑那方面。也就是说，他们创造出来的大量Java类都不可能轻易实现序列化。要给现有的类嫁接上序列化能力绝对是一种折磨。

即使Erlang4J能为Erlang term提供透明的序列化，问题还是会在Java与Erlang代码相接的地方冒出头来。对Java来说，不可序列化的对象是常规而非例外。

我不想单独针对某个项目，但太多人暗示Terracotta是可能的解决方案，所以我要特别解释一下为什么它是拙劣的替代品，至少和Erlang相比是这样。

第一，Terracotta是一个外部库，必定会强加一些限制。请看为了让Clojure运行在Terracotta之上要做多少工作[2]。Erlang的序列化和分布式能力是平滑无缝的，是深深植根在语言中的。

第二，Terracotta要经过配置才能正常工作。请到Terracotta网站上体会体会一个简单的例子要求多少配置[3]。我估计将Terracotta内建到语言运行时的确是可行的，但不可能在脱离大量配置的前提下办到。Erlang只需最低限度的配置——一个Cookie值，加上调用一次net_adm:ping/1，即已建立起可靠的分布式网络。Erlang的消息传递语义还简化了节点之间的通信，简化了对结点失效的处理。我不知道同样的话能不能用在Terracotta身上。

第三，Terracotta的“网络内存”模型代表的是共享一切的方案，与Erlang不共享任何东西的模型恰好是对立的。这里也一样，外部库做好了也免不了设下障碍，做坏了更会强加一种折中的总体设计。

观察：支持并发而不支持分布式编程削弱了系统整体的效用。换言之，能毫不费力地在单台机器上处理许多线程是很有意义的，但能让更多机器一起处理更多的线程及数据更有意义。

JVM不支持头等的闭包

JVM上的Erlang实现必须提供闭包，所以唯一的选项是在语言运行时中模拟闭包。Java和JVM确实提供了匿名内部类，可以作为闭包的近似。可是这种方式也有问题，因为它会产生更多的对象，最终都需要垃圾收集器回收，也就进一步恶化了单收集器瓶颈问题。

总结

本质上JVM的设计是为了高效率地运行OO语言。所以Ruby、Python这些语言（大体上）很容易移植到JVM平台。它们因此获得显著的性能提升也毫不令人意外，毕竟JVM平台是非常成熟和高质量的。

然而，JVM运行非OO语言就不那么合适。Erlang、Haskell和Scheme这类语言提供的一些特性，比如尾递归、闭包和Continuation，在JVM所针对的主流OO世界中都不是主角。它们与OO模型偏离得太远，以致JVM平台成了不明智的选择。

我相信我们正处在一次行业变革的发端，FP的远大前程才刚刚铺开。在我看来，新语言走上前台，也应该有新的平台随之流行才是自然的表现。■

参考资料

[1] SISC是一个基于Java的Scheme实现：http://sisc-scheme.org/

[2] Clojure + Terracotta集成报告：http://tinyurl.com/ClojureOnTerracotta

[3] Terracotta的HelloWorld例子：http://tinyurl.com/HelloClusteredWorld

[4] Reia是运行在BEAM虚拟机上的一种类似Ruby/Python的语言：http://reia-lang.org/

作者简介

Kevin Smith的开发生涯中做过许多大大小小的项目，甚至负责实现过两个Java标准：JSR-168和WSRP。由于从事大规模应用套件的性能调优工作，对JVM和GC有很深刻的理解。他从2006年起受到Erlang的感召，不但加入Engine Yard公司从事全职的Erlang工作，还为The Pragmatic Bookshelf制作了一系列很受欢迎的Erlang教学视频。现在他在自己的公司Hypothetical Labs工作。

■ 责任编辑：郭晓刚（guoxg@csdn.net）

利用OProfile对多核多线程进行性能分析

演示在采样分析工具OProfile的帮助下，如何一步一步地进行性能调优，最终使优化后的结果接近于理论值。

■ 文 / 杨小华

性能分析工具简介

在对应用程序不断调优的过程中，除了制定完备的测试基准外，还需要一把直中要害的利器——性能分析工具。

根据工具的复杂度和所提供的功能，可以将性能工具分为两个层次：

1. 基本的计时工具

在普通生活中，秒表是最简单的计时工具。根据该思想，可以将计时函数放在代码的任意位置并多次调用，这样就可以测量出整个应用或者某一部分的运行时间。这种分析方法不够精细，误差大。

2. 软件分析工具

目前，主要有两种不同类型的软件分析工具：采样和插桩。

- 采样型分析工具

主要通过周期性中断，来记录相关的性能信息，如处理器指令指针、线程id、处理器id和事件计数器等。这种方法开销小，精确度高。在Linux系统中，比较常见的有Oprofile和Intel VTune性能分析器等。

- 插桩型分析工具

既可以使用直接的二进制插桩，也可以通过编译器在应用中插入分析代码。这种方式与自己在应用中增加计时函数类似，同时带来的开销大，但提供了更多的功能，如调用树，调用次数和函数开销等。在Linux系统中，比较常见的有gprof和Intel VTune性能分析器等。

本文将利用采样型工具OProfile，对多核多线程程序进行性能分析，起一个抛砖引玉的作用。

衡量性能收益的方法

随着科学技术的不断发展，计算机系统结构朝着多核的方向发展，从而将并发编程推到了聚光灯下，但如何去衡量并行程序设计所带来的性能收益呢？

不得不想起1967年Gene Amdahl所作出的杰出贡献，他提出的Amdahl定律能够计算出并行程序相对于最优串行算法在性能提升上的理论最大值。

Amdahl定律，S表示执行程序中串行部分的比例，n表示处理器核的数量，H(n)表示系统开销：

$$加速比=\frac{1}{S+(1-S)/n+H(n)}$$

由于Amdahl定律本身做出了几个假设，但这些假设在现实世界中又不一定是正确的，因此使计算机界心灰意冷了很多年，认为根据Amdahl定律，开发更大的并行性所带来的性能收益可能是微不足道的，一直到Gustafson定律的出现，才改变了现状。

在Sandia实验室工作的基础上，E.Barsis提出了Gustafson定律，其中S表示执行程序中串行部分的比例，N表示处理器核的数量：

$$扩展加速比=N+(1-N)*S$$

幸运的是，Shi于1996年证明Gustafson定律和Amdahl定律是等效的。

OProfile工作原理简介

根据CPU系统结构的不同，OProfile支持两种采样方式：基于事件(Event Based)的采样和基于时间(Time Based)的采样。

如果CPU内部存在性能计数寄存器，则OProfile基于事件采样，记录特定事件（如分支预测事件）发生的次数，当达到设定的定值时就采样一次。反之，则基于时间采样，主要是借助于操作系统的时钟中断机制，每当时钟中断发生时就采样一次。不难看出，基于时间的采样方式，要求被测程序不能屏蔽中断，其精度也低于事件采样。

对于x86体系结构，不同型号的CPU，采样方式也不同，具体细节如表1所示。

表1　x86各处理器采样方式

处理器	CPU_TYPE	采样方式
Athlon	i386/athlon	Event Based
Pentium Pro	i386/ppro	Event Based
Pentium II	i386/pii	Event Based
Pentium III	i386/piii	Event Based
Pentium M (P6 core)	i386/p6_mobile	Event Based
Pentium 4 (non-HT)	i386/p4	Event Based
Pentium 4 (HT)	i386/p4-ht	Event Based
Dual Core	timer	Time Based
Core 2 Duo	timer	Time Based

OProfile主要分为两部分，其中一部分是内核模块(oprofile.ko)，另外一部分是用户空间的守护进程(oprofiled)。前者主要负责访问性能计数寄存器或者注册基于时间采样的函数，并将采样结果置于内核的缓冲区中。后者在后台运行，负责从内核空间收集数据，并写入采样文件中，其交互流程如图1所示。

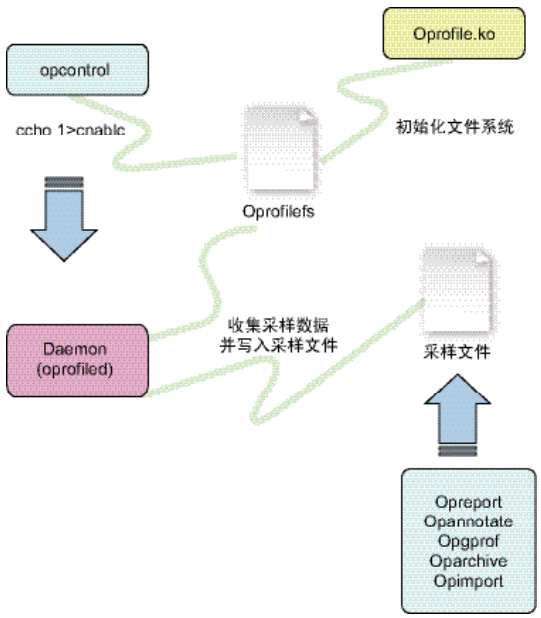


图1 OProfile交互流程图

安装OProfile

oprofile.ko内核模块已经被集成到Linux 2.6内核中，所以只需要安装前端工具，可以从OProfile官方网站下载源码来进行安装，当前最新版本为0.9.4。该源码包是通过automake和autoconf生成的makefile，所以只需要以root用户进入oprofile目录，运行./configure、make及make install来完成所有的安装。在目前大部分发行版中，安装时可能缺少popt库，请另行下载安装。

OProfile工具链提供了6大工具，供用户控制OProfile和分析样本。其中opcontrol是一个bash脚本程序，主要用来控制OProfile的启动、暂停及设置，其他工具主要是对采样数据进行分析。

样例程序

程序功能：求从1一直到APPLE_MAX_VALUE(100000000)相加累计的和，并赋值给apple的a和b；求orange数据结构中的a[i]+b[i]的和，循环ORANGE_MAX_VALUE次。

```
#define ORANGE_MAX_VALUE      1000000
#define APPLE_MAX_VALUE       100000000
#define MSECOND               1000000

struct apple
{
    unsigned long long a;
       unsigned long long b;
};

struct orange
{
       int a[ORANGE_MAX_VALUE];
       int b[ORANGE_MAX_VALUE];

};

int main (int argc, const char * argv[]) {
    // insert code here...
    struct apple test;
       struct orange test1;

       for(sum=0;sum<APPLE_MAX_VALUE;sum++)
       {
                test.a += sum;
                test.b += sum;
       }

       for(index=0;index<ORANGE_MAX_VALUE;index++)
       {
                sum += test1.a[index]+test1.
b[index];
       }
    return 0;
}
```

观察上述样例程序，*S*（串行部分的比例）所占比例非常小，基本为0，根据Amdahl定律，在双核系统（*n*=2）上，则加速比为：加速比＝1/(0+1/2+1%)＝1.96，也就是说效率可以提高96%。样例程序运行时间为：1.049046s，那么经过优化后，能不能达到理论值呢？请随着本文开始。

追踪热点

既然要充分利用双核的特性，则不得不对样例程序进行改造，进行并行程序设计。

可以将应用程序看成是众多相互依赖的任务的集合。将应用程序划分成多个独立的任务，并确定这些任务之间的相

互依赖关系，这个过程称为分解（Decomposition）。分解问题的方式主要有三种：任务分解、数据分解和数据流分解。

运用任务分解的方法，不难发现计算apple的值和计算orange的值，属于完全不相关的两个操作，因此可以并行。请看改造后的两线程程序：

```
void* add(void* x)
{
    for(sum=0;sum<APPLE_MAX_VALUE;sum++)
    {
        ((struct apple *)x)->a += sum;
        ((struct apple *)x)->b += sum;
    }
    return NULL;
}

int main (int argc, const char * argv[]) {
    struct apple test;
    struct orange test1={{0},{0}};
    pthread_t ThreadA;
    pthread_create(&ThreadA,NULL,add,&test);

    for(index=0;index<ORANGE_MAX_VALUE;index++)
    {
        sum += test1.a[index]+test1.b[index];
    }
    pthread_join(ThreadA,NULL);
    return 0;
}
```

OProfile的功能非常强大，可以对每个线程进行单独采样，也可以对每个CPU单独采样，这些都是通过opcontrol的separate选项来完成的，选项值含义如表2所示。

表2　separate选项值含义

separate选项值	含义
none	默认值
lib	对每个应用程序的所有lib进行采样
kernel	对每个应用程序的内核及内核模块采样
thread	对每个线程或任务采样
cpu	对每个CPU进行采样
all	以上所有选项的功能

操作步骤如下：

```
# opcontrol --init
# opcontrol --separate=thread --no-vmlinux
# opcontrol --start
Using 2.6+ OProfile kernel interface.
Using log file /var/lib/oprofile/samples/
oprofiled.log
Daemon started.
Profiler running.
# ./twothreadprocess
add thread:b7dc7b90,Used Time:1.225483
main thread:b7dc8ad0,Used Time:1.230151
a = 4999999950000000,b = 4999999950000000,sum=0
# opcontrol --shutdown
Stopping profiling.
Killing daemon.
```

识别出并行程序中的重载

运行opreporte，查看采样结果：

```
# opreport -l ./twothreadprocess
CPU: CPU with timer interrupt, speed 0 MHz
(estimated)
Profiling through timer interrupt
Processes with a thread ID of 5237
Processes with a thread ID of 5238
samples     %       samples       %      symbol name
30       100.000       0          0
main
0           0         343      100.000      add
```

运行时间为：1.230151s，与理论值相差甚远，反而运行时间越来越长，原因何在呢？

通过分析结果，不难看出add线程负载非常重，而main负载较轻，负载不均衡，因此重点分析对象为add线程。根据多线程数据分解的原理，将计算apple值的过程一分为二，main线程也参与部分计算。由于都是循环，为了使负载均衡，则add线程里面应该循环(ORANGE_MAX_VALUE+APPLE_MAX_VALUE)/2次。修改后的程序如下：

重新运行OProfile，再次收集采样数据，改造后的程序是否达到了负载均衡呢？

```
# opreport -l ./twothreadprofile
CPU: CPU with timer interrupt, speed 0 MHz (estimated)
Profiling through timer interrupt
Processes with a thread ID of 5242
Processes with a thread ID of 5243
samples  %        samples    %       symbol name
135    100.000       0       0
main
0         0         156   100.000         add
```

运行时间为：0.629821s，时间有了显著的提高，效率也提升了66%，已经逼近理论值，但还是有一点差距。

```
#define ORANGE_MAX_VALUE      1000000
#define APPLE_MAX_VALUE       100000000
#define MSECOND               1000000
#define MIDLLE_VALUE          49500000

struct apple
{
    unsigned long long a;
    unsigned long long b;
};
struct orange
{
    int a[ORANGE_MAX_VALUE];
    int b[ORANGE_MAX_VALUE];
};
unsigned long long value[2][2];

void* add(void* x)
{
    temp1 = ((struct apple *)x)->a;
    temp2 = ((struct apple *)x)->b;
```

```
    for(sum=MIDLLE_VALUE;sum<APPLE_MAX_VALUE;sum++)
    {
        temp1 += sum;
        temp2 += sum;
    }
    value[0][0]=temp1;
    value[1][0]=temp2;
    return NULL;
}

int main (int argc, const char * argv[]) {
    struct apple test;
    struct orange test1={{0},{0}};
    pthread_t ThreadA;
    test.a= 0;
    test.b= 0;
    temp1 = test.a;
    temp2 = test.b;

    pthread_create(&ThreadA,NULL,add,&test);
    for(sum=0;sum<MIDLLE_VALUE;sum++)
    {
        temp1 += sum;
        temp2 += sum;
    }
    value[0][1]=temp1;
    value[1][1]=temp2;

    sum=0;
    for(index=0;index<ORANGE_MAX_VALUE;index++)
    {
        sum=test1.a[index]+test1.b[index];
    }
    pthread_join(ThreadA,NULL);
    test.a = value[0][0]+value[0][1];
    test.b = value[1][0]+value[1][1];

    return 0;
}
```

继续分析，不难看出，负载还是没有达到均衡，main线程还是有点轻。因此继续增大MIDLLE_VALUE的值到53500000，再次运行OProfile，看看效果如何：

```
# opreport -l ./twothreadprofilebig
CPU: CPU with timer interrupt, speed 0 MHz
(estimated)
Profiling through timer interrupt
Processes with a thread ID of 5248
Processes with a thread ID of 5249
samples  %        samples    %          symbol name
145    100.000       0       0             main
0         0          146   100.000         add
```

运行时间为：0.540942s，比样例程序时间大约减少了一半，效率提升了92%，已经非常接近于理论值。相信通过不断的微调，将会越来越接近理论值。同时在计算理论值时，所假设的系统开销比较低，仅仅为1%。

由于Linux内核进程调度器天生具有CPU软亲和力（affinity）的特性，这就意味着进程通常不会在处理器之间频繁的迁移。在实际应用中，如果不存在数据竞争的影响，应用的不同部分分布到不同的CPU上运行，可能会带来更高的收益。

将样例程序的多线程版本绑定到不同的CPU上运行，效率会有所提升吗？

修改后的程序如下：

```
struct apple
{
    unsigned long long a;
    unsigned long long b;
};

struct orange
{
    int a[ORANGE_MAX_VALUE];
    int b[ORANGE_MAX_VALUE];
};

int cpu_nums = 0;
unsigned long long value[2][2];

inline int set_cpu(int i)
{
    CPU_ZERO(&mask);
    if(2 <= cpu_nums)
    {
        CPU_SET(i,&mask);
        if(-1 == sched_setaffinity(gettid(),sizeo
f(&mask),&mask))
        {
            return -1;
        }
    }
    return 0;
}

void* add(void* x)
{
    if(-1 == set_cpu(1))
    {
        return NULL;
    }

    temp1=((struct apple *)x)->a;
    temp2=((struct apple *)x)->b;

    for(sum=MIDDLE_VALUE;sum<APPLE_MAX_VALUE;sum++)
    {
        temp1 += sum;
        temp2 += sum;
    }
    value[0][0]=temp1;
    value[1][0]=temp2;

    return NULL;
}

int main (int argc, const char * argv[]) {
    struct apple test;
    struct orange test1={{0},{0}};
    pthread_t ThreadA;

    test.a= 0;
    test.b= 0;
```

```
    cpu_nums = sysconf(_SC_NPROCESSORS_CONF);

    if(-1 == set_cpu(0))
    {
        return -1;
    }
    temp1=test.a;
    temp2=test.b;

    pthread_create(&ThreadA,NULL,add,&test);

    for(sum=0;sum<MIDDLE_VALUE;sum++)
    {
        temp1 += sum;
        temp2 += sum;
    }
    value[0][1]=temp1;
    value[1][1]=temp2;

    sum=0;
    for(index=0;index<ORANGE_MAX_VALUE;index++)
    {
        sum+=test1.a[index]+test1.b[index];
    }

    pthread_join(ThreadA,NULL);
    test.a=value[0][0]+value[0][1];
    test.b=value[1][0]+value[1][1];

    return 0;
}
```

修改opcontrol的separate参数为cpu，然后开始采样：

```
# opcontrol --separate=cpu --no-vmlinux
# opcontrol --reset
# opcontrol --start
Using 2.6+ OProfile kernel interface.
Using log file /var/lib/oprofile/samples/
oprofiled.log
Daemon started.
Profiler running.
# ./affinitytwoprofile
......
# opcontrol --shutdown
Stopping profiling.
```

采样结果如下：

```
# opreport -l ./affinitytwoprofile
CPU: CPU with timer interrupt, speed 0 MHz
(estimated)
Profiling through timer interrupt
Samples on CPU 0
Samples on CPU 1
Samples on CPU all
samples       %          samples     %
samples       %           symbol name
147           100.000       0             0
147      50.9559          main
0                  0             143      100.000
143      49.0441          add
```

程序运行时间为：0.575131s，与没有绑定之前的效果接近，但不代表CPU绑定没有用处，只是本样例程序运行时间较短，工作量不大，不适合使用CPU绑定而已。

OProfile分析多核程序与分析多线程程序类似，通过采样数目来识别重载，然后开始一步一步地优化。

总结

根据以上分析及实验，对所有改进方案的测试时间做一个综合对比，如图2所示.

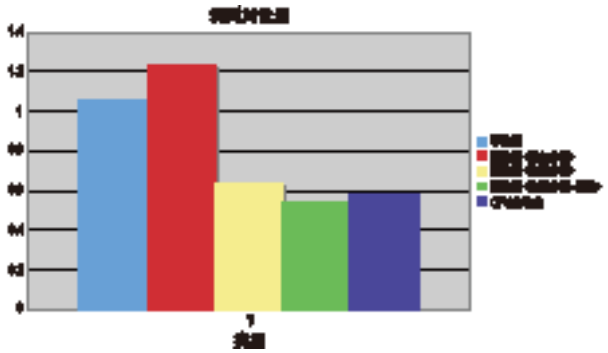


图2 各优化时间对比图

利用OProfile，一步一步地不断调优，最终使优化后的结果接近于理论值，让我们见证了优化工具所具有的魅力。但OProfile不仅仅只有这些功能，关于更多的其他功能，请参见官方网站介绍或者本文参考资料所列出的资料2和资料3。■

参考资料

[1] OProfile 官方网站

[2] PrPrasanna S. Panchamukhi，《用OProfile彻底了解性能》，IBM DeveloperWorks

[3] John Engel，《使用OProfile for Linux on POWER识别性能瓶颈》，IBM DeveloperWorks

[4] 杨小华，《利用多核多线程进行程序优化》，IBM DeveloperWorks

作者简介

杨小华，某通讯设备公司软件工程师，从事嵌入式Linux方面的开发和研究。

■ 责任编辑：郭晓刚（guoxg@csdn.net）

安全编码实践之五

地址空间格局随机化ASLR

■ 文 / 褚诚云

概述

在前面安全编码实践中我们介绍过GS编译选项和缓存溢出，以及数据保护DEP。首先，缓存溢出的直接后果就是可能导致恶意代码的远程执行，于是编译器提供了GS保护。但是，GS选项有自身的局限，存在若干方法可以绕过GS选项的保护。于是进一步，操作系统提供了数据执行保护，即DEP，以及与之对应的NXCOMPAT编译选项。

那么，是不是现在我们就可以高枕无忧了？在安全领域中，系统的攻防是一个不断发展进化的过程。DEP提出后，就出现了针对DEP的Ret2libc攻击手段。这一点我们曾在介绍DEP的安全编码实践文章的最后简单提及过。

ASLR（Address Space Layout Randomization），地址空间格局的随机化，就是用来防范Ret2libc攻击手段的另一个重要的安全特性。那么，什么是Ret2libc攻击，ASLR的原理是什么，开发人员如何使用这个安全特性，就是我们这篇文章要探讨的内容。

DEP和Ret2libc攻击

DEP对堆栈溢出的保护

在栈溢出介绍中提及到Windows体系结构下函数堆栈布局（地址从高向低）如表1所示。

表1 Windows系统的函数堆栈结构

调用参数
返回地址
EBP上层函数堆栈基址
异常处理代码入口地址 （如果函数设置异常处理）
局部变量

如果发生堆栈溢出，恶意代码通过覆盖在堆栈（stack）上的局部变量，从而修改函数的返回地址，而导致恶意代码执行。下面是这类攻击方式的堆栈结构的一个典型例子。

表2 堆栈溢出时的堆栈结构

调用参数	覆盖方向—>	恶意代码
返回地址		恶意代码的入口地址
EBP上层函数堆栈基址		溢出的变量覆盖区域，往往包括必要的填充字节
异常处理代码入口地址 （如果函数设置异常处理）		
局部变量		

当DEP保护机制被使用后，由于恶意代码是存放在系统的数据页面（堆栈页面上），那么函数返回时，指令寄存器EIP将跳转到恶意代码的入口地址。此时该页面是非可执行的（non-executable），于是DEP就会触发系统异常而导致程序中止。

Ret2libc攻击

在上述的DEP保护机制中，可以看到关键是在函数返回时EIP跳转到了非可执行页面时被DEP检测到。那么Ret2libc的攻击原理是，攻击者设定的函数的返回地址并不直接指向恶意代码，而是指向一个已存在的系统函数的入口地址。由于系统函数所在的页面权限是可执行的，这样就不会触发DEP异常。

那么，攻击者应该将EIP控制指向哪个特殊的系统入口函数？一个例子是在Unix 系统下，libc是一个共享的C动态执行库，里面有许多非常有用的函数，例如system函数。它的定义如下：

```
int system(const char *string);
```

函数system()可通过运行环境来执行其他程序，例如启动Shell等。那么，攻击者就可以通过构造以下的堆栈结构[1]。

表3 Ret2libc攻击的堆栈结构

调用参数	覆盖方向—>	/bin/sh
		虚假的返回地址
返回地址		system函数的入口地址
EBP上层函数堆栈基址		溢出的变量覆盖区域，往往包括必要的填充字节
异常处理代码入口地址 （如果函数设置异常处理）		

这样，当发生堆栈溢出的函数返回时，EIP跳转到system函数。因为system函数本身就是可执行的，这时不会产生DEP异常。攻击者通过构造system函数的调用参数来可以启动其他程序。在攻击过程中，函数返回到libc库（return to libc）是关键，这也就是Ret2libc名字的来由。

细心的读者也许已经发现，在表3中，没有任何恶意代码被插入。攻击者虽然可以通过system或者其他系统函数来执行很多敏感的操作，但在多数情况下，还是更希望可以执行自身定制的恶意代码。如何可以做到这一点？于是在最初的Ret2libc的攻击方式的基础上，又发展出特别针对Windows系统攻击的手段。它的原理是通过VirtualProtect函数来修改恶意代码所在内存页面的执行权限，然后再将控制转移到恶意代码。

VirtualProtect是Windows系统kernel32.dll提供的函数，其功能是修改调用进程所在虚拟地址空间（virtual address）的内存区域的保护权限。它的定义如下：

```
BOOL WINAPI VirtualProtect(
  __in   LPVOID lpAddress,
  __in   SIZE_T dwSize,
  __in   DWORD flNewProtect,
  __out  PDWORD lpflOldProtect
);
```

攻击者构造以下的堆栈结构[2]来调用VirtualProtect。

表4 使用VirtualProtect攻击的堆栈结构

调用参数	覆盖方向——>	恶意代码
		lpflOldProtect值
		设定可执行权限参数
		恶意代码页面的大小
		恶意代码所在内存页面的基址
		恶意代码的入口地址
返回地址		VirtualProtect函数的入口地址
EBP上层函数堆栈基址		溢出的变量覆盖区域，往往包括必要的填充字节
异常处理代码入口地址（如果函数设置异常处理）		
局部变量		

首先，当发生堆栈溢出的函数返回时，EIP跳转到VirtualProtect函数。注意到这里攻击者特别构造将恶意代码的入口地址作为VirtualProtect函数退出时的返回地址。由于在VirtualProtect的执行过程中，恶意代码所在的页面被修改为可执行权限，这样当VirtualProtect返回时，EIP再跳转到恶意代码时就不会触发任何DEP异常。

除了使用VirtualProtect函数，攻击者还可以使用其他函数，例如NtSetInformationProcess等。

ASLR和/dynamicbase链接选项

在上面对Ret2libc攻击方式的介绍中，我们看到最为关键的一点是攻击者事先预知了特定函数，如system或VirtualProtect的入口地址。在Windows XP或Windows 2000上，这些函数的入口地址是固定的，即攻击者事先可以确定的。

在Windows Vista中引入了ASLR安全特性。它的原理就是在当一个应用程序或动态链接库，如kernel32.dll，被加载时，如果其选择了被ASLR保护，那么系统就会将其加载的基址随机设定。这样，攻击者就无法事先预知动态库，如kernel32.dll的基址，也就无法事先确定特定函数，如VirtualProtect的入口地址了。

ASLR是系统一级的特性。系统动态库，如kernel32.dll，加载地址，是在系统每次启动的时候被随机设定的。

下面是一个简化的ASLR演示程序[3]。

```
// aslr.cpp : Demo the dynamic base of DLLs due to
ASLR
//

#include "stdafx.h"
#include <windows.h>
#include <stdio.h>

void foo( void )
{
      printf( "Address of function foo = %p\n",
foo );
}

int _tmain(int argc, _TCHAR* argv[])
{
      HMODULE hMod = LoadLibrary( L "Kernel32.
dll" );
      // Note—this is for release builds
      HMODULE hModMsVc = LoadLibrary( L"MSVCR90.
dll" );

      void* pvAddress = GetProcAddress(hMod,
"LoadLibraryW");
      printf( "Kernel32 loaded at %p\n", hMod );
      printf( "Address of LoadLibrary = %p\n",
pvAddress );
      pvAddress = GetProcAddress( hModMsVc,
"system" );
      printf( "MSVCR90.dll loaded at %p\n",
hModMsVc );
      printf( "Address of system function =
%p\n", pvAddress );

      foo();

      if( hMod ) FreeLibrary( hMod );
      if( hModMsVc ) FreeLibrary( hModMsVc );
      return 0;
}
```

这段程序的目的是输出kerner32.dll和msvcr90.dll的基址，loadlibrary和system函数的入口地址，以及应用程序本身一个函数foo()的入口地址。

使用ASLR非常简单。从Visual Studio 2005 SP1开

始，增加了/dynamicbase链接选项。/dynamicbase选项可以通过Project Property -> Configuration Properties -> Linker -> Advanced -> Randomized Base Address，或直接修改linker的命令行编译选项即可。见图1。

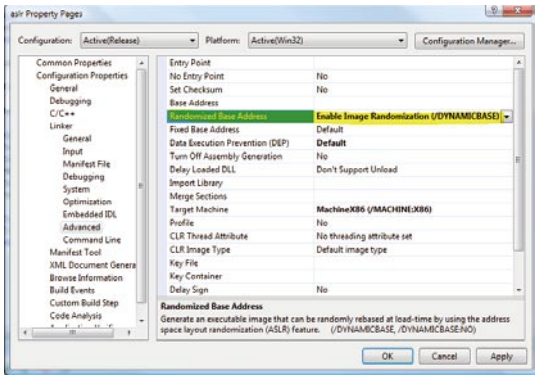


图1 /dynamicbase链接选项配置

在Visual Studio 2008环境，用Win32 Console Application类型，编译链接演示程序。注意，如果使用Visual Studio 2005 SP1的话，需要将msvcr90.dll更改为msvcr80.dll。

如果程序没有使用ASLR功能的话，在Windows Vista下运行。输出的结果是：

```
Kernel32 loaded at 763F0000
Address of LoadLibrary = 7641361F
MSVCR90.dll loaded at 671F0000
Address of system function = 6721C88B
Address of function foo = 00401800

重启系统

Kernel32 loaded at 76320000
Address of LoadLibrary = 7634361F
MSVCR90.dll loaded at 6A340000
Address of system function = 6A36C88B
Address of function foo = 00401800
```

我们看到，即使程序本身没有使用ASLR，Kernel32.dll和MSVCR90.dll的加载地址也发生了变化。这是因为这两个库都已经选择了被ASLR保护。但是应用程序自身foo()函数的地址是固定的。

如果程序使用ASLR功能的话，在Windows Vista下运行。输出的结果是：

```
Kernel32 loaded at 763F0000
Address of LoadLibrary = 7641361F
MSVCR90.dll loaded at 671F0000
Address of system function = 6721C88B
Address of function foo = 003B1800

重启系统

Kernel32 loaded at 76320000
Address of LoadLibrary = 7634361F
MSVCR90.dll loaded at 697A0000
Address of system function = 697CC88B
Address of function foo = 00871800
```

应用程序自身函数foo()的加载地址也随着系统重启发生了变化。即一旦使用了/dynamicbase选项，生成的程序在运行时候就会受到ASLR机制的保护。

ASLR的局限

首先，ASLR安全特性只在Windows Vista和其后的Windows版本（如Windows Server 2008）中实现。

其次，ASLR是需要和DEP配合使用的。如果CPU不提供对于DEP的硬件支持，或者应用程序没有选择被DEP保护的话，恶意代码一旦执行，就可以通过程序进程表结构来获得特定DLL的加载基址。

就性能和兼容性而言，ASLR的实现上都做了考虑，没有太多的影响。一个范例是Microsoft Office 2007。Office 2007的程序全面使用ASLR功能，并没有发现对其性能和兼容性带来太大的影响[4]。

总结

ASLR安全特性在Windows Vista和其后的Windows版本（如Windows Server 2008）中实现。它可以防范基于Ret2libc方式的针对DEP的攻击。ASLR和DEP配合使用，能有效阻止攻击者在堆栈上运行恶意代码。建议开发人员使用/dynamicbase链接选项让开发的应用程序或动态链接库使用ASLR功能。■

参考文献

[1] Bypassing non-executable-stack during exploitation using return-to-libc，http://www.infosecwriters.com/text_resources/pdf/return-to-libc.pdf，c0ntex

[2] A Brief History of Exploitation Techniques & Mitigations on Windows，http://hick.org/~mmiller/presentations/misc/exploitation_techniques_and_mitigations_on_windows.pdf，Matt Miller

[3] Writing Secure Code for Windows Vista，Michael Howard, David LeBlanc

[4] Use of ASLR, NX, etc，http://blogs.msdn.com/david_leblanc/archive/2008/03/14/use-of-aslr-nx-etc.aspx，David LeBlanc

作者简介

褚诚云，微软Windows安全部门反病毒组的软件开发工程师。2001年1月加入微软美国总部。先后申请多项数字版权保护及反病毒技术专利。近年来在微软技术大会上主讲过多个有关信息安全的技术讲座。中文Blog站点（http://blogs.itecn.net/blogs/chengyun_chu/）

■ 责任编辑：郭晓刚（guoxg@csdn.net）

主持人：张银奎
《软件调试》一书作者，英特尔亚太研发中心高级软件工程师。从事软件开发和研究十余年，对IA-32架构、操作系统内核、虚拟技术，尤其对软件调试有较深入研究。翻译（合译）作品包括《数据挖掘原理》、《机器学习》、《人工智能：复杂问题求解的结构和策略》等。

拯救挂死的PowerPoint

这一期给大家讲一个真实的故事，起源是笔者亲身经历的一次应用程序挂死，主要内容是探寻挂死原因的求索过程，末尾是关于“谁之错”的思考。因为很想早点与大家分享这个故事，所以把上个月开始的托管系列中断一期（抱歉无法一一征求读者同意）。

戛然而止

故事发生的时间是3月8日，因为笔者没有权利享受妇女节，于是只好伏在电脑面前“做功课”。这天要完成的任务是使用PowerPoint准备一份讲义，内容是关于调试的——我最喜欢的话题。启动PowerPoint，打开惯用的模板，想出一个喜欢的题目——《用户态崩溃和转储分析》，列出简单的提纲，然后选取几年来积累的资料，驾轻就熟，很快就做好了二十几个页面；当然还应该取个中意的文件名，保存一下。很快几个小时便过去了，我仍然不觉得疲倦，正如古人所说的：“我自乐此，不为疲也”，眼看着再有几页就做完了。这时需要把一个脚本文件插入到PPT中，为的是在演讲时可以打开真实的脚本来讲解。于是我便把PowerPoint的窗口调小一些，用鼠标抓起脚本文件，移动到PowerPoint窗口的投影页面，鼠标箭头右下方出现预期的加号，松开鼠标释放……但是释放后，我很快感觉到了异常，“货物”没有落在页面中，鼠标箭头右下方的加号还在，这意味着拖动的“货物”没有放下去。点击PowerPoint窗口的图标，就像点到木头上，过了几秒钟再点时，窗口标题中出现了熟悉的“(Not Responding)”（图1），PowerPoint挂死了，刚才还灵活自如的窗口元素现在都僵硬不动了，喧嚣归于宁静，一切戛然而止……

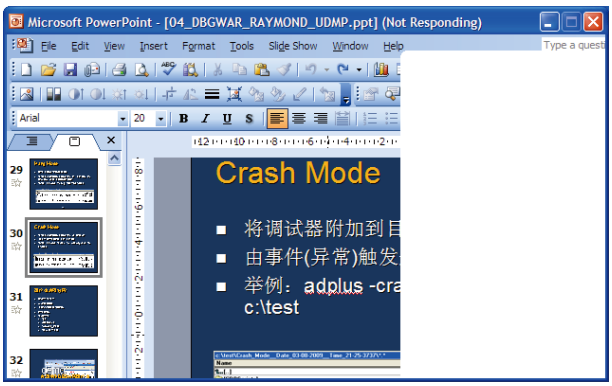


图1 PowerPoint戛然而止

捶胸顿足

宁静了几秒钟，我狠狠地一跺脚，“哎呀，还有好几页没有保存呢！”怎么办？所有按钮和菜单无法操作，想通过简易的方法把文件保存起来几乎不可能了。把进程杀掉？杀掉简单，但是没有保存的东西很可能就永远不见了。虽然前面曾经保存过，但是最近十几分钟所做的内容还没有保存；虽然十几分钟的劳动不值几个钱，但是也舍不得白白放弃呀；虽然Office程序有文件自动保存和恢复机制，但是没人确保它一定工作呀；虽然……更何况，自己是开发软件的，又是热衷调试的，并且说过不畏软件难题，对于这样的问题怎么能轻易放过呢，是可忍，孰不可忍？

拍照存档

首先使用ADPlus给PowerPoint程序和从中拖出文件的Windows Commander程序“拍照”，也就是将这两个进程的当前状态转储到文件中。使用的命令如下：

Adplus -pn powerpnt.exe -pn wincmd32.exe -hang -o c:\test

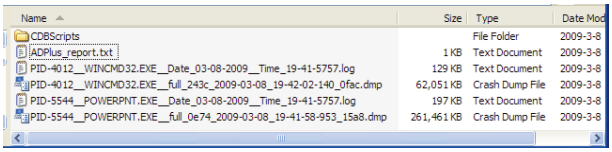


图2 使用ADPlus产生的转储文件和日志报告

命令执行后，便会产生一系列转储文件、日志文件和报告文件（图2），这样便保存下了永久的现场资料，即使短时间无法找到问题，那么还可以等以后有时间或者时机成熟时继续分析，让问题“躲过初一，躲不了十五”。

上调试器

调试器是解决复杂软件问题的最重要工具（《软件调试》第6篇序），检验这句话的时候又到了。启动WinDBG，附加（File > Attach to a Process…）到执行映像为POWERPNT.exe的进程，稍稍停顿后，WinDBG成功附加到目标，列出进程内的所有模块后，中断下来，等待命令。

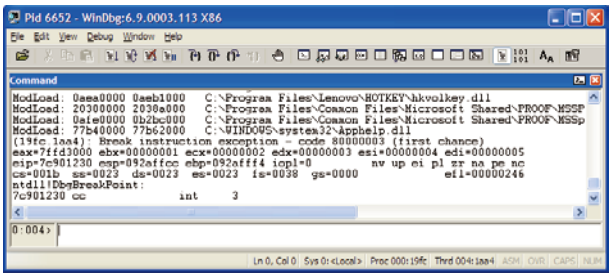


图3 将WinDBG附加到挂死的PowerPoint进程

先浏览一下进程内各个模块，看是否有可能是病毒的异常模块侵入。没有发现可疑对象，接下来该如何分析呢？

时光倒流

软件调试中的栈回溯（Stack Backtrace）技术可以根据保存在栈上的信息生成一个线程的函数调用过程，因为是从当前的执行点反向追溯父函数，所以叫栈回溯。对于眼下的问题，界面完全失去响应，这说明负责消息处理的界面更新的UI线程阻塞了。对于几乎所有Windows GUI程序，编号为0的初始线程就是UI线程。因此，使用~0s命令切换到0号线程：

```
0:004> ~0s
eax=0b490000 ebx=774e1a3c ecx=00000000
edx=7c90eb94 esi=000003e0 edi=000003e0
eip=7c90eb94 esp=00139d54 ebp=00139d7c iopl=0
nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b
gs=0000             efl=00000202
ntdll!KiFastSystemCallRet:
7c90eb94 c3              ret
```

在上面显示的寄存器信息中，eip代表程序指针寄存器，它的值代表着CPU将执行的指令，更确切地说，是CPU下次执行这个线程时要执行的指令。它目前值所对应的符号是：ntdll!KiFastSystemCallRet，也就是一条ret指令（函数返回）。这意味着，当前线程已经进入到了内核态执行，用户态的上下文中保存的是从内核态返回后将要执行的指令和状态。

执行kn 100命令显示栈回溯，100表示要显示的深度：

```
0:000> kn 100
 # ChildEBP RetAddr
00 00139d50 7e4194ae ntdll!KiFastSystemCallRet
01 00139d7c 775c00be USER32!NtUserMessageCall+0xc
02 00139d9c 775c24ce ole32!wInitiate+0x3e
03 00139db8 775bda92 ole32!CDdeObject::CProxyMana
gerImpl::Connect+0x5f
04 0013a1f4 775bdd00 ole32!CDdeObject::DocumentLe
velConnect+0x3f
05 0013a638 775f9904 ole32!DdeBindToObject+0xde
06 0013a8c0 7757c79c ole32!CPackagerMoniker::Bind
ToObject+0xc7
07 0013a8ec 775c9ad3 ole32!BindMoniker+0x60
08 0013a908 775c9edb ole32!wCreateFromFileEx+0x1f
09 0013a988 775ca0d0
ole32!OleCreateFromFileEx+0xfb
0a 0013ac24 7757becb ole32!wCreatePackageEx+0x198
0b 0013ac68 7757bf1f
ole32!OleCreateFromDataEx+0xea
0c 0013aca4 3028ad21 ole32!OleCreateFromData+0x42
WARNING: Stack unwind information not available.
Following frames may be wrong.
0d 0013acec 302c419a POWERPNT+0x28ad21
0e 0013ad28 303ddf42 POWERPNT+0x2c419a
[为节约篇幅略去多行]
24 0013dcec 7e418806
USER32!InternalCallWinProc+0x28
25 0013dd54 7e4189bd USER32!UserCallWinProcCheckW
ow+0x150
26 0013ddb4 7e418a00 USER32!DispatchMessageWorker
+0x306
27 0013ddc4 30034f03 USER32!DispatchMessageW+0xf
28 0013dde8 30034eca POWERPNT+0x34f03
29 0013ddf8 30034cf5 POWERPNT+0x34eca
2a 0013de40 30004a7b POWERPNT+0x34cf5
2b 0013ff18 30004a2c POWERPNT+0x4a7b
2c 0013ffc0 7c816ff7 POWERPNT+0x4a2c
2d 0013fff0 00000000
kernel32!BaseProcessStart+0x23
```

扫描一眼这个栈回溯结果，最下方是kernel32!BaseProcessStart函数，即初始线程正式开始执行时的起点，其上的几行（#28~#2c）都显示为POWERPNT加一个很大的偏移值，这意味着没有找到POWERPNT模块的符号，所以只好以模块的起始地址为参照物，微软的符号服务器

包含了大多数Windows系统模块的公开符号文件，但是没有包含Office程序的符号文件，因此这样显示也属正常。栈帧#0c~#09中的函数与OLE（对象链接与嵌入）有关，栈帧#05~#03中都包含DDE（动态数据交换）字样，这些信息与挂死前的文件拖动操作很吻合。栈帧#01表示发起一个系统服务调用NtUserMessageCall，栈帧#0表示以快速系统调用方式进入到内核态。

从栈回溯可推测出初始线程因为NtUserMessageCall这个系统调用而进入到内核态执行，“至今”尚未返回。事实上，大多数应用程序挂死也都是“挂”在内核态。

顺藤摸瓜

要想了解挂死的更多原因，就要进一步分析挂死前的执行过程，也就是继续从上面的栈回溯中挖掘线索。很多函数都使用栈来传递参数，因此可以尝试通过观察参数取值来搜集更多资料。要观察参数，最好知道参数类型，这又需要知道函数原型。因为我们没有系统模块的私有符号文件，所以没有办法让WinDBG直接显示函数原型和参数（kPL命令）。怎么办呢？因为SDK中公开的函数在MSDN中有原型公布，所以可以从公开的API下手。纵观栈帧中的各个函数，OleCreateFromData、OleCreateFromDataEx和OleCreateFromFileEx都是公开的API，查看MSDN，OleCreateFromFileEx的第二个参数是字符串类型：

```
HRESULT OleCreateFromFileEx(
  REFCLSID rclsid, //Reserved; must be CLSID_NULL
  LPCOLESTR lpszFileName, //Pointer to name of file
to initialize
```

于是可以根据对应栈帧的基地址（0013a988）得到第二个参数的地址（EBP+C），然后用dU命令来观察参数值：

```
0:000> dU poi(0013a988+c)
001cc458  "c:\dumps\dde\Hang_Mode__Date_03-"
001cc498  "08-2009__Time_16-22-1818\CDBScri"
001cc4d8  "pts\PID-1076__INETINFO.EXE_IIS_I"
001cc518  "n-Process_Applications.cfg"
```

恰恰是当时拖动的文件的完整路径，看来PowerPoint正在接收文件，但意外挂死了。Google一下或者看一下SendMessageW的反汇编，就可以知道SendMessage API内部会调用NtUserMessageCall这个内核服务，而且两者的函数原型也相似的。显示有关栈帧的参数：

```
0:000> kb 2
ChildEBP RetAddr  Args to Child
00139d50 7e4194ae 7e441396 00010014 000003e0
ntdll!KiFastSystemCallRet
00139d7c 775c00be ffffffff 000003e0 001d06dc
USER32!NtUserMessageCall+0xc
```

NtUserMessageCall的第一个参数是要发送消息的目标窗口句柄，而0xffffffff具有特殊含义，是用来广播消息的（HWND_BROADCAST）。第二个参数是要发送的消息，因为SDK中并没有把所有Windows消息常量定义在一起，所以寻找0x3e0所代表的消息名称不是特别简单，我们不妨先放下这条线索。

分析到这里，可以推测出，PowerPoint的初始线程正在做接收拖动过来的.cfg文件的工作，在完成这个任务的过程中，它调用了OleCreateFromFileEx API，后者内部调用了DDE有关的内部类和方法，而DDE的方法出于某种目的而调用内核服务NtUserMessageCall开始广播0x3e0消息。而这个消息广播调用一去不回。

上下求索

既然UI线程进入内核态执行迟迟不归而导致用户界面失去响应，那么很自然地想到去看看这个线程在内核态干什么？

于是再运行一个WinDBG实例，开始一个本地内核调试会话（File > Kernel Debug > Local）。然后执行下面的命令找到PowerPoint进程：

```
lkd> !process 0 0 powerpnt.exe
PROCESS 88bfbb80  SessionId: 0  Cid: 19fc    Peb:
7ffd3000  ParentCid: 03c4
    DirBase: 18900e00  ObjectTable: e44e0090
HandleCount: 268.
Image: POWERPNT.EXE
```

然后列出这个进程的各个线程结构（以下输出信息的格式做过调整）：

```
lkd> !PROCESS 88bfbb80 2
PROCESS 88bfbb80  SessionId: 0  Cid: 19fc    Peb:
7ffd3000  ParentCid: 03c4
    DirBase: 18900e00  ObjectTable: e44e0090
HandleCount: 268.
    Image: POWERPNT.EXE
        THREAD 891c4020  Cid 19fc.1dbc  Teb:
7ffdf000 Win32Thread: e2456b00 WAIT: (Suspended)
KernelMode Non-Alertable SuspendCount 1
FreezeCount 1
```

然后执行!THREAD 891c4020命令显示第一个线程（UI线程）的详细信息（图4）。

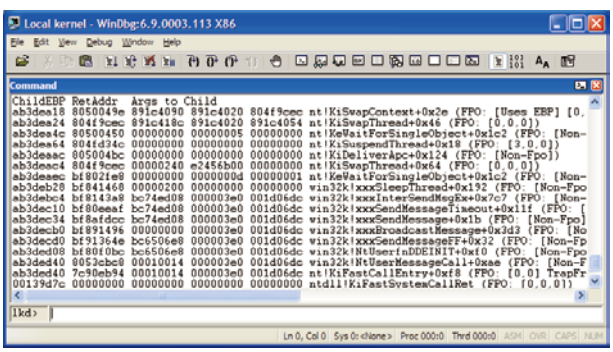


图4 UI线程在内核态执行的过程

观察图4所示的内核态栈回溯，从很多函数名中包含的SendMessage字样就可以知道这个线程正在内核态执行发送消息的任务，这与前面的分析是一致的。

现在的关键是找到向哪个窗口发送消息时停滞不前了。这还是需要分析参数，可以看到第二个参数那一列中有多行都是3e0，说明是在发送3e0这个消息。再看xxxSendMessage、xxxSendMessageTimeout和xxxInterSendMsgEx这个几个函数的第一个参数，它们的第一个参数都是bc74ed08，这个值很可能与发送消息的目标窗口有关。或许是包含窗口句柄的指针，使用DD命令观察它：

```
lkd> dd bc74ed08 l1
bc74ed08  001506c4
```

如何验证001506c4是不是一个窗口句柄呢？有多种方法，可以使用Spy++工具来查找，也可以使用Skywing编写的sdbgext扩展模块中的hwnd扩展命令来观察：

```
0:000> !sdbgext.hwnd 001506c4
Window    001506c4
Name
Class     WindowsForms10.Window.0.app.0.3ce0bb8
WndProc   00000000
Style     WS_OVERLAPPED
ExStyle   WS_EX_WINDOWEDGE WS_EX_LEFT WS_EX_
LTRREADING WS_EX_RIGHTSCROLLBAR
HInstance 00400000
ParentWnd 00000000
Id        00000000
UserData  00000000
Unicode   TRUE
ThreadId  000016d8
ProcessId 00001c68
```

果真是窗口句柄，该窗口所属的进程ID和线程ID分别是00001c68和000016d8，转化成十进制（使用.formats命令），然后到任务管理器中查找，就可以看到进程的名称了。

幕后黑手

分析到这里，我们定位到了阻碍PowerPoint的UI线程的窗口句柄和所属进程，很可能是这个进程不回复PowerPoint发过来的消息而导致PowerPoint阻塞在那里。那么这个进程为什么不回复消息呢？运行第三个WinDBG实例，附加到这个进程，使用~*命令列出所有线程，然后切换到线程号为000016d8的线程（~0s），观察栈回溯：

```
0:000> kn 100
 # ChildEBP RetAddr
00 0012f0a0 7c90e288 ntdll!KiFastSystemCallRet
01 0012f0a4 7c801875 ntdll!NtReadFile+0xc
02 0012f10c 77deb3cb KERNEL32!ReadFile+0x16c
03 0012f138 77deb25f ADVAPI32!ScGetPipeInput+0x2a
04 0012f1ac 77deb568
ADVAPI32!ScDispatcherLoop+0x3f
05 0012f40c 00a6a762 ADVAPI32!StartServiceCtrlDis
patcherW+0xe3
WARNING: Frame IP not in any known module.
Following frames may be wrong.
06 0012f428 67a241d2 0xa6a762
07 00000000 00000000 System_ServiceProcess_
ni+0x41d2
```

从上面的栈回溯可以看出，这是一个使用.Net语言编写的系统服务，0xa6a762是即时编译后产生的代码所在的位置。从栈帧#05~#00可以看出这个线程进入了等待系统服务控制命令（暂停、停止服务等）的循环，在等待服务管理器发送给它的命令。也就是说它根本没有等待和处理窗口消息。概括一下，这个进程创建了一个顶层的窗口，但是没有合适的窗口循坏，因为没有对系统发送给它的消息作出合适的分发和处理。

起死回生

找到了导致问题的进程和原因后，在系统服务管理器中停止这个服务，然后将附加到PowerPoint进程的WinDBG分离开（Debug > Detach Debuggee），这时会发现，PowerPoint起死回生了，又焕发了活力，所有功能也都是正常的。

环境问题

回过头来看这个问题，到底是谁的错呢？如果说是PowerPoint，似乎它的实现也没有明显的错误，它按部就班地调用API，也不知道发送消息会得不到回复，事实上，其他在UI线程这样做的程序也有这个问题。如果说是哪个系统服务程序的问题，它也可以争辩“我为什么一定要处理你的消息？”“不处理，你就应该死在那儿吗？”听着也有道理，事实上，如果用调试器将某个窗口程序中断下来，也会造成类似的影响。推而广之，很多挂死的问题都是因为双方或者多方协作时出了问题。前面提到过的那个3e0消息是WM_DDE_INITIATE消息，这是16位的Windows所定义的消息，当时整个系统的任务调度机制都是协作方式的，要求系统中的每个任务谦恭礼让，有绅士风度，但是…… 扯远了，就此打住，下一期再见！■

本期问题：

除了本期讨论的情况，你知道还有哪些原因导致应用程序挂死吗？

编者说明：

- 投稿信箱：contest@csdn.net
- 解答提交时间，最好能早于当月15日。
- 联系方式写在一个单独的TXT文件里，包括以下几项：
1）姓名　2）工作单位或学校　3）电话联系方式
4）邮寄地址　5）E-mail地址

■ 责任编辑：郭晓刚（guoxg@csdn.net）

BI让企业更“聪明”

“Business Intelligence”这个词，有些人听过，有些人没有。面对BI，我们有欣喜，有泪水。大家都寄希望于BI能使企业更加“聪明”，但有时候却事与愿违。本期月度关注将带您走进BI的世界，帮您揭开诸多有关BI的疑团。

商业智能是端到端的解决方案

——访微软中国研发集团战略合作部首席商业智能架构师朱宁

■ 记者 / 欧阳

2005年，Gartner的一项调查显示商业智能（BI）是在所有CIO心中第二重要的系统。而从2006年开始一直到2008年，众多的CIO都一致认为BI是未来最值得关注的领域。但是在大多数人眼里，BI却始终概念不清。在BI领域奋战了十多年的微软商业智能架构师朱宁认为，首先要澄清的，就是BI的概念。

BI到底是什么？

朱宁认为：“很多人之前并没有意识到BI的价值。随着它越来越重要，对BI的理解也开始慢慢成熟。原来很多人认为这是一个高高在上的东西，只是一些高深的人才能做的应用系统。而一般人根本无从涉及，毕竟这涉及到很多智能的领域。另一些人则认为BI就是报表，就是展现在人们面前的那些图标。很明显这些理解都并不完全正确。很多人可能如我多年前刚遇到BI一样，认为今天被那些大厂商收购的企业，比如Cognos、BO这样的企业就是BI，但今天我却认为，这些企业的技术和产品，在整个BI的系统栈里只是一个前端的展示应用而已。由于他们有很多能帮助用户查询数据做分析的产品，所以很多人就理所应当地认为这些企业就是做BI的。但如果这些技术和产品就算是BI的话，那么就应该是这些公司来占有整个BI的市场份额，而不是IBM、Oracle、SAP和微软这四架马车。”

为了进一步解释到底哪些产品和企业是做BI的，而哪些只是在这个领域承担了一部分工作，朱宁从系统的角度分析了BI。

“首先，对于任何一个系统来说，必须要有一个数据源。通常是数据仓库，要有数据集成的部分。面对各种各样不同的数据源，一个统一版本的数据事实至关重要。以前很多决策都是拍脑袋，但现在不同，我们必须将决策建立在事实的基础之上。这是一个很大的挑战：我们拥有大量的数据，但真正从数据里拿到自己想要的信息却并不容易。比如可口可乐，比如耐克，它们都知道自己的数据在一个大的数据库里。但面对一些类似这样的问题：我的十大客户情况如何？我的库存控制应该怎样？哪些区域卖的很好？那些产品有比较好的市场反响？尽管可以做到，但是这需要大量的专业程序员进行大量编码，从而把数据从黑洞中取出来。等到这些信息提取真正完成的时候，黄花菜都凉了。尽管一些企业的前端展现做得很酷，但如果其背后没有一个性能很好的数据仓库来支撑，并通过中间环节进行信息的深加工，那么就绝对不能适应这种情况。”

企业级，准备就绪

那么BI服务的对象呢？作为一出生就定义为帮助企业进行决策的系统，BI肩负的任务格外沉重。对于微软来说，尤其如此。

毕竟企业级和桌面市场是有很大不同的。微软要将自己的产品在企业级领域准备好，其实是一个很大的跨越。朱宁解释道："我加入微软的团队就叫企业级解决方案部门，那个团队里的其他成员也有很多是来自原来本身就做企业级产品的公司。我曾帮助耐克、波音、世界银行等等一大批企业级的客户设计其BI架构。当然，这些都是基于微软技术来做的。特别是SQL2005，加强了很多对企业级BI客户至关紧要的功能，使微软在BI领域中突飞猛进。"

回过头来看这几年BI领域的大举收购。其他企业级的厂商，包括IBM、SAP和Oracle，都拿到了大量企业的数据，但如何从黑洞中取出信息，并能够用这些信息为客户决策提供依据，依然是一个关键问题。恰巧BO、Cognos、Hyperion等做BI前端的企业能够较好地完成这项工作，因此市场的收购开始变得越来越多。

构建BI系统栈

然而，用技术人员的方式理解BI，就会知道：对于任何一个完整的系统而言，采用今天的B/S架构一定有相同的三层结构，有前端展现层，有中间业务逻辑层，也有后端数据层。BI也不例外。但从产品上来说，原来的IBM与Oracle都拥有其自身的后端数据层。随后包括DB2 9以及Oracle 11g，都对非结构化数据有了很好的支持。但从BI产品的角度上来说，却未免有失完善。

从这个角度看，就知道为什么IBM会收购Cognos，Oracle会收购Hyperion了。由于只有整个系统栈的后端数据层，因此在中间层上需要有合适的产品来补充，这其中也包括前端的展现层。因为只有这样才能真正意义上完成BI整个解决方案的系统栈。微软的产品从SQL Server 2000开始就在向这个方向靠拢，因此它可以说是唯一一个开始就在构建整个BI系统栈的厂商。"在这方面，我们面临的整合问题要小得多。"朱宁解释道。

"比如从关系数据库来说，三大数据库厂商都各有千秋。但在多维数据库领域，也就是在中间层，微软的优势却要远远大于对手。以产品销售为例，在关系数据模型里，我们能够很容易地比较我们的产品销售情况与去年相比如何，然而一旦要加上地域性地销售数据，那么这个数据内容立刻就变成二维的空间了，再加上产品销售与市场推广的关系，与销售人员的素质关系等维度，你会发现每一个决策的制定都是多维的过程。关系数据模型只有二维，因此做这件事就非常困难，或者说可以做但效率极低。因此中间层的价值立刻就得到了非常有效的体现。"作为技术人员的朱宁谈到产品的核心技术显得更加热情。


微软商业智能架构师朱宁，一位自学成才的资深专家

BI关键在行业

然而，无论BI的系统栈如何搭建，关键问题仍在于完整解决方案要解决的问题。对于BI来说，其核心目标是为了帮助用户进行商业决策，因此其着眼点一定是在行业的业务和经验上。这个观点朱宁非常赞同，并认为这是对BI核心任务非常到位的理解。

几个在BI领域中常常被当作谈资的话题通常都在零售行业。由于零售行业的特殊性以及消费用户的消费习惯关联性，BI能够非常直接地为企业决策带来快速的回报。往往一个策略的制定和执行，立马就能给销售数据带来很大提升。在这个过程当中，需要业务专家对消费市场非常熟悉，并了解当前情况下的主要趋势与方向。同时，BI在另外一些行业的应用，甚至具备更高的价值。

"比如微软内部有专门的医疗行业团队，对这个行业的发展和趋势进行深入研究。我在中国一年多的时间内，接触了很多电信、金融、保险等行业的用户，但坦白说，从微软早期的产品战略上看，对这些行业的理解还并不是非常到位。这是由于微软是一个以平台和技术为核心的企业，我们要做的是尽力打造好平台与技术，而对行业的理解，则更多由微软众多的合作伙伴来完成。经过近10年时间的发展，微软已经有非常完整的商业生态系统了，通常情况下与客户的合作都有非常重要的合作伙伴参与，他们是行业的专家，这样也就形成了良好的相互依赖关系。"

结束语

在微软的眼里，BI仍然是一个具有巨大潜力的市场。但这个市场目前还远远没有到达成熟阶段。四大玩家虽然各有特色，但微软依然会沿着自己做基础技术平台的道路前进。而有志于在BI领域发展的开发商，毫无疑问将面对历史的机会，一旦能够坚持到BI进入主流市场，将会迎来一个温暖的春天。■

■ 责任编辑：李雨来（liyl@csdn.net）

云计算在企业信息建设和商务智能领域的应用

■ 文 / 窦蒙

作为信息系统中结合业务主题、基于数据内容发现并指导业务的关键领域，商务智能的重要性主要体现在三个方面：

- 促进企业决策流程；
- 降低整体营运成本；
- 协同组织目标与行动。

不过，在拉平的世界中商业机会从“孕育”到“凋谢”的周期越来越短，市场规则已从“大鱼吃小鱼”过渡到“羚羊饿死大象”。这时候，我们需要审视自己的商务智能发展策略是否还能一味朝着“大型”发展？

商业机会从“发现”到“领域商业机会结束”的间距越来越短，从最初的几十年、几年到现在的几天、几小时甚至就是当前OLTP的同时。

举个例子，一位长者到银行取了5万，我们的立方体之前没有关于其家庭成员的维度，这样我们可能就忽略了该信息而让用户从营业大厅离开了。如果我们在与另一家保险公司的合作中（当然这只是个例子，具体操作中这涉及隐私信息泄露）获得长者的投保受益人信息——他的儿子，而他的儿子之前在银行办过商业贷款。这些信息分散在各处，而且之前并没有很好的规则可以提取并“清洗”它们，但从操作层面来讲，我们完全可以把它们串在一起，让柜台人员当时就推荐一些更适用的业务产品。

反观我们的商务智能系统，由于一般开发周期比较长，经常在“领域商业机会结束”之后才上线，但此时不免晚了；即便有些商务智能项目可以提前上线，但无论对于分析还是发掘可能积累的数据样本规模都不够，无法出现中国版本的“尿片与啤酒码放规律”，到头来还可能对商业行为产生错误的决策支持。

商务智能的三类

另一方面，以往的商务智能往往将用户定位到企业的高层管理者和专业的业务数据分析师，但对于当前恶劣的市场环境，我们要照顾好一线的市场和销售人员。也许在经济困难时期，我们应该适当调整商务智能也就是要提供面向企业全员，尤其要照顾好一线这些直接为企业赚钞票的员工。这样，根据应用目的不同，我们将商务智能划分为三类：

- 战略型；
- 战术型（也往往被称之为分析型）；
- 操作型。

如果把一个企业比作汽车，他们之间的关系就好像：

- GPS导航系统决定汽车的行驶方向；
- 汽车仪表盘面板指示汽车各类关键指标；
- 油门、刹车、轮胎上的传感器。

操作型商务智能由于其时效性特点，可以用于Call Center在线附加销售推荐、检修附加服务、客户电邮联系中的邮件签名广告、保险附加产品宣传等类似的领域，在完成主要业务推荐的过程中，最大限度提供用户可能最需要的附加服务，进而提升销售业绩。它的典型特点如下：

- 以每小时、每日、隔日、周内数据为基础；
- 服务于Real-time、Right-time的业务；
- 数据来源分散：包括企业自身、合作伙伴、云等来源；
- 其间，可能存在大量非结构化信息，以及没有被统一“清洗”过的信息。

商务智能与云计算

作为同样热点的技术领域，云计算也成为很多企业，尤其是大型企业（虽然也内普遍认为云计算适合中小企业，但笔者反而认为对于经常受到IT制肘的“大象”型企业而言，更具商业意义）扩充业务外延的主要手段，由于这个领

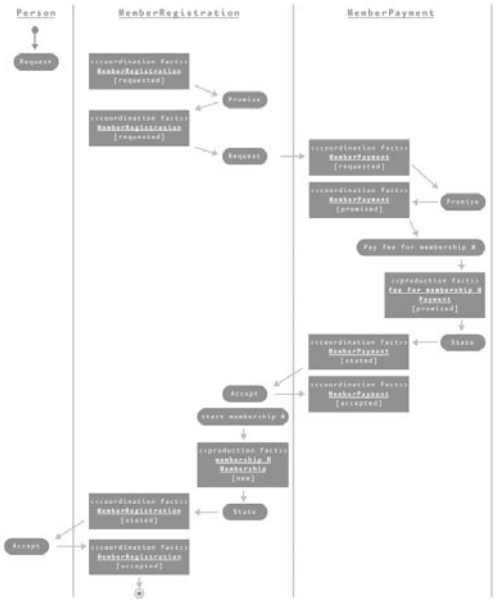


图1 电子银行业务处理流程图

域的实践不多，因此很多云应用还停留在在线业务交互阶段，但我们不难发现

图2 BI的三类

这种将信息以及使用信息的软件同时发布到云的计算模式可能会带来更多的用户，考虑到现阶段云计算的收费特点，采用操作型商务智能也许是一个比较现实同时也比较容易见效的方式。因为，云应用可以帮助企业做到“三个更”：

- 更大：按需付费方式满足更加庞大的企业用户群体；
- 更少：减少软件、服务器、存储、人员的不确定投入；
- 更好：借助云服务商对于大型IT系统的管理经验和最佳实践，保证云应用始终在SLA约束下提供持续有保证的IT服务交付；

配合操作型商务智能，企业可以帮助作为企业应用外延的云应用的部分发掘更大规模用户中的潜在商业价值。那么对于“更少”和“更好”又如何实现呢？作为特殊的云应用，基于笔者之前一些云项目的实施经验，基于云信息的操作型云商务智能应用需要做一些特殊处理。不过在此之前我们要先了解当前主要的云计算收费方式：

- 基于网络流量的计费；
- 基于存储容量的计费；
- 基于计算时间的计费。

因此，我们可以根据上面的收费方式对云商务智能做如下处理：

- 更多地采用XML数据：现阶段主要云厂商提供的开发接口提供的都是KeyValuePair Collection或者是Entity Model的数据访问接口，如果这些预申报或者预录入的数据还采用关系方式设计，那么当我们针对“当前”这个业务数据进行处理的时候发起一个Join操作，真正提取一个业务实体的时候，其成本会比较大，加之云平台存储的虚拟特点，计算时间也比较长。因此，不妨从一开始就将整个业务实体以XML方式保存，操作型商务智能的各种规则、商业预警信息（比如：一个语义网络）、阀值也以XML方式存储，针对命中业务实体的智能指导信息也以XML方式作为业务实体子树的方式合并后一次性呈现给交互用户；
- 由于现阶段还没有基于云的商用分析服务，因此在进行数据钻取、发掘算法计算的时候，往往会产生一些中间结果，出于同样的目的，我们也可能需要把他们临时持久化为XML信息保存在云存储中；
- 更多采用Update而非Insert方式。对于企业内部应用而言，我们往往习惯于“记流水帐”，将所有操作全部Insert追加，但考虑到存储成本关系，我们可能会将一些没有必要长时间留存的指标、阀值、规则采用Update方式。

举个例子而言，对于一个竞价系统，它将预录入信息前端处理部分开发为一个云应用，部署在云端，其中包括一个最简单的价格瞒骗分析功能，需要动态对于某细类商品的价格高低位价格进行登记。

对于传统的INSERT方式可能会像下面这样存储：

```
PID    FROM     TO       MAX   MIN
-------------------------------------------------
120013 2009-01-16 2009-01-25 20 15
120013 2009-02-16 2009-02-25 22 17
120013 2009-04-16 2009-04-25 22 15
12003  2009-01-16 2009-04-25 20 15
```

但对于跑在云上的应用来说，这样需要花费更多的存储费用，所以基于Update的方式可以只保留当前操作型商务智能需要的这个信息。比如上面的信息，在2009-01-21的时候看到结果可能是：

```
PID    MAX   MIN
-------------------------------------------------
120013 20    15
12003  20    15
```

在2009-04-21时看到结果又可能是：

```
PID    MAX   MIN
-------------------------------------------------
120013 22    15
12003  20    15
```

另外，为了尽量减少网络流量成本、减少计算时间，我们可以对云服务、云数据库、云BPM、云队列、云应用针对业务需要打包，一次性部署到云端，计算时相关组成尽量在云服务商平台完成交互。例如图1是一个位于云中的电子银行业务处理流程：

如果每次交互都基于最低颗粒度的云应用服务接口，那么需要多次网络流流量费用，而将他们打包为一个部署在云上的粗颗粒度“支付”服务，则可以减少很多额外开销。

最后，部署上考虑到很多大型企业存在分支机构、卫星团队（Satellite Team）的特点，可以将操作型商务智能以及它需要服务部署成下图的样子：

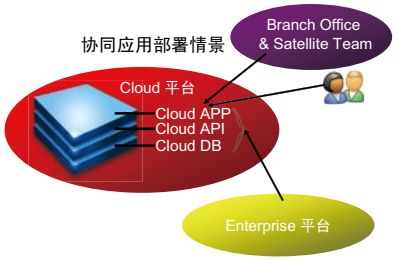


图3 操作型商务智能部署情景

参考文献

QConf 2009 云计算专题《企业协同商务智能设计》演讲，演讲者：王翔。

■ 责任编辑：李雨来（liyl@csdn.net）

数据为王
——记 IBM 眼中的商业智能

■ 特邀记者 / 许舟平

当漫天飞舞的都是“3G让上网搜索更加简单”、“3G让即时通讯随时随地”、“3G让手机办公轻松自如”的中国电信天翼“带您畅游3G”广告语的时候，中国移动的高层们并不为之所动。尽管中国电信的天翼品牌已经拉到了诸如李开复、丁磊和MSN中国区总裁Erik Johnson等人，为其开始了新一轮3G广告的狂轰滥炸，但是在今天国内移动通信市场已经进入一个相对稳定的“胶着”阶段时，拥有大量的忠诚而有商业价值的客户，才是每一个电信运营商的获胜之道。而如何选择、标识乃至获得并维护最有价值的客户，为这些高端客户提供优质服务并最终获得高收益，在中国移动看来，一切都将要用数据说话。IBM的商业智能正帮助中国电信业做着这一切。

IBM软件部信息管理（Information Management）负责数据仓库和商业智能的技术顾问李磊先生介绍说，目前在IBM拥有成熟全套的数据仓库/商业智能产品，涵盖数据仓库引擎，ETL工具，多维分析，前端工具，数据挖掘，系统管理，应用服务器等各个方面。其中出色的OLTP和OLAP能力的是海量数据仓库引擎是其解决方案的亮点。在李磊看来，数据仓库系统也必然要和其它业务系统互连，所以具有优异的互联能力、集成性能，同时要对主流的硬件平台和操作系统做支持是必不可少的。而随着企业发展，一个完善的数据仓库系统必须具有良好的线性扩展的能力。具有了这些特性的IBM数据仓库系统，对于使用者来说，在海量的数据中进行数据挖掘和处理，就如同探囊取物一般。

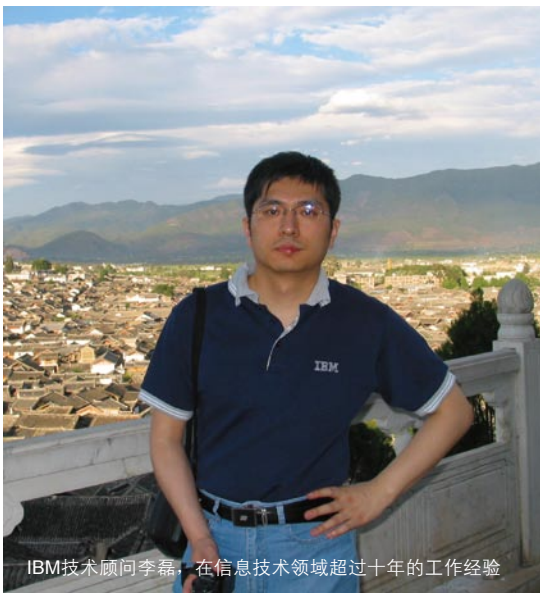
李磊举了一个某省移动公司的例子来说明IBM的商业智能。该移动的钻石卡、金卡和银卡客户人数虽然仅占总用户数很少的比例，但其带来的收入却很高，是单位客户收益最高的客户。为了为这些高端客户提供优质服务，锻造高端服务品牌，该移动公司专门成立了包括钻石卡、金卡和银卡客户在内的客户俱乐部，并设立专门的服务部门－俱乐部客户服务部，并在其数据仓库上构建了一个面对俱乐部客户的维系应用。而面对联通、电信的节节紧逼，如何在现有市场基础环境下，保持该移动会员俱乐部的客户稳定，进一步改善俱乐部客户服务、提升俱乐部客户价值，则成为该移动俱乐部客户服务面临的主要问题。李磊谈到：其实该移动具体来说可细化为3个问题。

1. 地毯式服务，缺乏针对性

2. 粗放式服务，资源难以有效整合

3. 一线营销服务人员缺乏系统级智能化支撑，营销能力和效率有待提升，包括只有不完整的客户数据，缺少客户全景信息与客户知识，造成了“盲人摸象”式的客户认知；数据与工作流程之间缺少互动，后台的数据没有很好地与前台的工作流程整合。后台数据与分析无法有效传递到前台从而无法有效支撑其工作。前台工作中所产生的数据与经验也无法沉淀从而指导后续工作。

面对这些问题，在IBM的商业智能看来，该移动已拥有大量的客户和业务数据，只要对这些数据进行深入的数据分析和挖掘，一切都可以唾手可得。具体的内容包括：



IBM技术顾问李磊，在信息技术领域超过十年的工作经验

客户细分：细分专题是进行深度客户洞察的基础专题。通过细分专题，将看起来都一样的高端客户按照客户的业务行为模式和价值进行合理区分，使其业务特征鲜明，目标精确。北京移动将高端客户合理细分成3大类14个各具特色、重点突出、特征独立的细分人群，推进“大众地毯式”服务模式转变为针对目标人群的差异化服务模式。

流失预警：流失预警专题支撑起与KPI紧密相关的高端客户流失管理，通过流失预警专题及时发现高危客户，提前进行关怀维系工作。

套餐匹配：发现即使提前找到高危客户，也难以决策如何挽留维系，为此研发了套餐匹配专题，为需要关怀维系的客户找到合适的资费套餐，有效将其维系和捆绑，避免其流失。

俱乐部客户个体维系系统：随着应用的不断增加和深入，平台化的俱

乐部客户个体维系系统应运而生了，系统同时支持面向分析人员的群体分析，任务管理和面向执行人员的个体分析和客户维系工作管理。系统和前期建设的各个专题有机整合，最终形成一个体系化的俱乐部客户精细化服务平台，全方位支撑起某运营商高端客户服务。

在这些应用中，我们不难看出数据仓库在IBM商业智能中的重要性。按照以前的经验，IBM按照实施的时间周期以及应用的强壮性的递增，总结出有4种主要的实施数据仓库的方法：基于业务系统数据简单的汇总，单独数据库服务器用于分析，单一数据集市和数据仓库。虽然每种实施的策略各有其优缺点，但是数据仓库的解决方案在所有的实施方案中，是最稳定最强建的。

数据仓库是实施商业智能的一种方式。建立在数据仓库之上的商业智能应用，可以帮助用户更为准确的，及时地进行决策。数据仓库不是技术，是一种解决方案。关于数据仓库的概念，在李磊看来，有以下几点针对数据仓库的看法需要正确理解：

1. 数据仓库只是把数据堆积在一起?

在数据仓库实施的初期，有些商业智能项目只是把原来业务系统的中的业务数据复制到其它单独的数据库中，数据结构和定义和原来系统一致，在此之上在建立商业智能应用。这样的系统，最终用户很快就会发现问题：系统的维护和使用困难，系统的性能急剧下降。

作为数据仓库，首先要有企业级的数据仓库模型。数据仓库模型是针对商业智能分析应用的特点的，是业务专家在总结厂商的实施经验的基础上构建的，按主题存储的模型结构。

来自各个业务系统的各种类型的数据，通过ETL完成数据的整合、清洗、加载，保证进去数据仓库的数据是干净的、一致的。

2. 数据仓库等于低性能?

出于分析的要求，数据仓库都会保留历史信息，随着时间的积累，数据量会非常大。比如，IBM在国内的最大数据仓库的数据量已达三百TB。在这种情况下如何保证查询和ETL的效率是很大的技术挑战。

一个好的数据仓库解决方案，必定会考虑商业智能分析的特点，存储各种粒度的数据，满足分析和性能的要求。比如：IBM数据仓库模型中，会包括细节数据区、汇总数据区、星型结构数据区、系统维护数据区等。这样虽然花费了很少的额外的存储空间，但这样会极大地提高数据仓库的性能。

3. Stage区设计

一般的，来自业务系统的数据首先会进入到数据仓库的Stage区，该区域的数据，只保留很短的时间，一般只是保留前一天的数据，在此基础上，再进行清洗、转换等工作。

很多的情景下，ETL是对多表进行JOIN操作，然后再装载到数据仓库中。只有将业务系统的数据，抽取到Stage区，才能进行JOIN操作。

将数据抽取到Stage区，如果ETL程序出现问题，可以从Stage区开始进行数据的重新清洗、转换，而不用从源系统的数据抽取开始。

也可以满足某些特殊的查询。

4. ODS的含义

按照IBM对ODS的定义：

a) ODS是面向主题的，比如面向客户或者产品主题，而不是面向应用的。

b) ODS是集成的，是把面向主题的数据，汇集、集成的一个映像。比如，如果该ODS是面向客户主题的，那么和客户相关的信息都可能会作为ODS的一部分。

c) ODS是实时的，它反映了数据源系统的当前的状态。可能不会存储历史数据，历史数据会存储在数据仓库的相应区域内。

由于，ODS会实时地反映源业务系统的变化。那么在不同时间对ODS的查询会不一致。这也称为ODS的挥发性。

d) ODS存储的是细节数据。

5. 数据仓库进行系统设计时，要充分考虑性能问题。

正如前面提到的，数据仓库不等于低性能。不过由于数据仓库的数据量巨大，很容易出现性能下降问题。

例如，直接针对数据仓库的查询可能有多个人多次重复进行，比如银行业务中对损益科目数据的查询。类似的应用应该在系统设计的时候，就要考虑到。而不是把所有的压力都要强加到DB上。

6. 元数据管理问题

随着数据仓库的建立，建立在其上的商业智能应用也会越来越多。几百上千张表，以及无数的抽取流程以及流程之间的依赖关系，都会让数据仓库管理员头疼。所以必须充分利用元数据管理工具，有效的对元数据进行管理。

7. 数据仓库只是用来查询的?

现代数据仓库技术和原来的比较，有了更大的发展。比如实时性的要求，数据以插入方式进入到数据仓库中，数据仓库内的数据整理，会有大量的更新操作，数据仓库不仅提供报表，多维分析，同时也以服务方式被另外系统使用，以消息方式提供数据等。

在IBM的商业智能策略中，拥有一个好的数据仓库系统是远远不够的，针对于不同行业的不同的情况，应用不同的体系架构结合IBM系统的自身特点，来帮助客户完成各种需求，才是IBM能够在竞争激烈的商业智能市场取得领先的法宝。例如，在电信、银行业由于数据仓库系统构建比较系统化，在联机分析，数据挖掘等方面都有应用，所以IBM的商业智能系统就主要关注分析系统如何和业务系统配合，如何支撑营销，如何能够与业务流程融合在一起，形成业务优化的闭环。■

■ 责任编辑：李雨来（liyl@csdn.net）

BI 融合之道

■ 文 / 徐懿

在传统的信息科技归类中，Business Intelligence（BI）经常是和数据仓库关联在一起的。很多时候，BI就被当做数据仓库的前端展现工具来看待。人们在选择一款BI产品的时候，主要关心其OLAP（联机分析处理）引擎的性能是否优越，报表的格式是否丰富，即席查询工具是否易用等等。

Oracle自从2007年收购Hyperion以来，则更多地把BI和EPM（企业绩效管理）关联在一起，原因是Oracle有丰富的应用解决方案，而这些应用解决方案的用户都需要用商务智能来做决策支持。

Oracle以BI冠名的产品有BI Applications和BI Suite Enterprise Edition Plus等等。前者覆盖了财务、人事、采购、供应链、销售、服务、市场等企业的各个业务线（Line of Business）的分析型应用；而后者则提供了传统上狭义的BI的主要功能：报表发布、即席查询和交互性的仪表盘。

但是Oracle的BI解决方案远远不只限于这两个产品。事实上，Oracle提供了业界最完整的BI解决方案，如图1所示。

其中，Enterprise Performance Management应用主要是从Hyperion收购的产品，它支持目标设定、建模、计划、监控、分析和报表编制这样一个完整的财务管理周期。它和BI Applications构成了整个BI解决方案的上层建筑。并且利用BI Foundation来集成来自多个数据源的数据和提供信息仪表板、报表及分析功能。

Oracle Essbase是BI Foundation的重要组成部分之一，也是业界最快的多维OLAP引擎。它提供了建模和高级分析功能，例如可以进行“What-if”分析、模拟复杂场景和预测业务。它和BI Suite EE Plus里的BI Server可以双向集成，既可以作为BI Server的数据源，也可以作为目标。在最底层的数据仓库，Oracle数据库不仅提供了关系型的OLAP引擎和数据挖掘软件，还有分区功能支持超大规模数据仓库。这些都是大家已经熟知的。

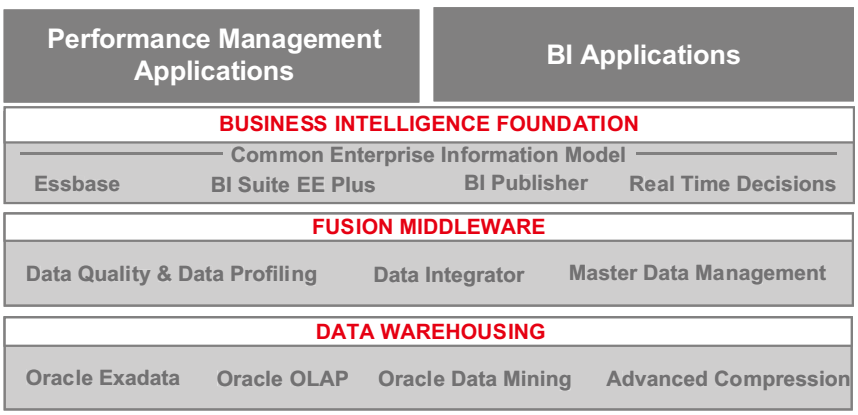


图1 Oracle BI解决方案

现实中的BI

随着互联网的应用模式日益渗透到企业的日常运营中，市场对BI的需求也在不断地变化。几周前，笔者和国内某大型网游公司的CTO交谈时，对方曾提到，在每款网络游戏的运营过程中，都必须有强大的用户行为分析系统支撑。如果不能及时掌握用户对某件新的游戏装备或道具的心理需求，就无法保证游戏的盈利。因为如果盲目提高了某种热门道具的供应量，很可能带来的不是更多的销售额，而是道具的迅速“贬值”，因而浪费了它的价值。这是个很有意思的话题，因为它揭示了新一代BI的特点之一：实时性。

另一个很有典型意义的例子是淘宝网的BI系统。目前每天在淘宝网上进行的交易量超过3亿元，订单数目超过200万，如此庞大的交易量既便是在全球C2C网站中也是排在前列的。淘宝网基于Oracle 网格计算技术，用12台小型服务器构建了价廉物美的数据仓库。这个数据仓库不仅为淘宝网自身的运营提供分析服务，还成为向淘宝网上成千上万的店主提供收费的BI服务的基础，内容包括多角度地分析各种商品受关注程度，分析来访用户的访问页面和所在地区等。淘宝网的新模式有两个方面的意义：首先，企业建立数据仓库不再只有一个昂贵的选择，还可以有更实惠的按需建设

的方法，从这个角度来说，很多企业都可以提前启动BI系统的建设，边使用边扩展；其次，不但企业自身的运营决策可以收益于BI系统，还可以把BI系统作为产生营收和利润的来源。

新一代BI的特点

从上面的两个例子我们可以大致地看到新一代BI的几个特点：

第一，业务系统与BI系统是紧密集成的。以前业务系统和BI系统互相剥离，出发点是避免影响业务系统的性能。但是新模式下的BI系统往往需要业务系统的大部分数据，而不是一小部分汇总数据。而新技术的采用，使得BI系统直接和业务系统集成成为可能，这一点可以在后面的Exadata的介绍中印证。

第二，BI系统所使用的数据是海量的。不仅是历史数据的量很大，新模式下的BI系统每天可能新增的数据量也很大。有统计表明，平均每过两年，数据仓库的数据量就会增长三倍。

第三，BI系统的实时性要求提高了。现在的企业做决策的时间窗口越来越小，尤其是在全球金融危机的大环境下，企业的敏捷性更加重要。

第四，BI系统新的模式不单单是从业务系统到BI系统的单向流动，BI系统产生的数据也会向业务系统流动，从而影响业务的新走向。

第五，BI系统必须支持异构的数据源。由于BI系统和业务系统的集成更加紧密，BI系统往往直接从多个不同的业务系统里抓取数据。这就面临多个不同的数据库，甚至是多个不同厂商的数据库。而数据定义的不统一会导致数据分析和挖掘的无效性。

这些新特点对BI系统的体系架构提出新的要求。实时性要求这个体系架构必须保证实时的数据查询，响应时间足够短。和业务系统的集成，双向性和异构性要求这个体系结构必须是开放的，可以与其它技术进行交互。海量数据要求这个体系结构必须能够提供从服务器端向存储端延伸的网格扩展能力。而这些因素集合在一起又要求BI系统的数据准备过程要比传统的ETL（抽取-转化-装载）更加高效和灵活，而且需要能在不同数据源之间对数据定义做统一的管理。

融合之道

因为这些新的要求，Oracle的BI解决方案和Oracle的中间件、应用一样，也采取融合的策略。

在今天的Oracle BI解决方案中，不再只是传统的BI Suite EE和Essbase。首先，Oracle在数据仓库上进一步加强了对BI的支持能力，最显著的就是去年刚发布的Oracle Exadata服务器和Oracle HP Database Machine。

要解决数据量高速增长，存储和服务器内存之间出现瓶颈的问题，不能光靠增加硬件的投入。如果仅仅将存储模块化以增加更多的通道，以及用Infiniband等更高速的通道取代传统的光纤通道，仍然不能完全解决瓶颈问题。要是能结合一种软件的方法，减少需要传输的数据量的话，那么传输的瓶颈就可以被彻底地打破了。Oracle Exadata就是结合了以上三种方案，软硬兼施的解决方案。根据Exadata用户的反馈，使用Exadata可以提高查询性能高达10倍以上，有些场景甚至可以达到70倍以上。

而Oracle HP Database Machine结合了已经渐渐成为主流的Oracle RAC(真正应用集群)技术和Oracle Exadata技术，实现了网格计算的数据仓库服务器。这就满足了BI系统体系结构的实时能力和从服务器延伸到存储端的网格扩展能力。

其次，在Oracle最新的11g数据库中，增加了新的特性支持海量数据。11g里引入的Advanced Compression支持对应用程序透明的数据压缩，使得存储空间可以节省2到4倍。结合从10g数据库就有的Partitioning、OLAP和Data Mining，Oracle数据库强大的数据仓库功能使其成为全球数据仓库市场份额最高的数据库。

再次，Oracle Data Integrator也成为Oracle BI解决方案的一个组成部分。它可以从异构的各种业务系统数据源中抽取数据，并高效地导入Oracle数据仓库中，供Oracle BIEE Plus生成各种报表或各种查询。Oracle Data Integrator完成的就是从传统的ETL(抽取-转化-装载)向更高效的ELT(抽取-装载-转化)的转换。结合Data Quality和Data Profiling，Data Integartor可以保证数据整理的质量。

最后，Oracle将Master Data Management Suite作为BI系统的组成部分之一，在快速变化的动态业务环境中实现主数据管理，保证在不同系统之间进行数据整合时的数据一致性。

我们看到，融合已经成为Oracle的BI解决方案的一个特色。Oracle从分析型应用，到BI基础，到中间件，再到数据仓库的大融合使得商务智能从少数企业和少数人的特权应用变成了更多企业能够用得起，更多用户能受益的普惠应用。而笔者完稿之时，刚听到Oracle购并Sun的消息，这使得Oracle的产品线覆盖了“从应用到磁盘”。Oracle的BI融合之道，可能很快又要开始新的章节了。■

作者简介

徐懿，Oracle大中华区产品战略部资深解决方案专家。1996年至2003年他分别在Oracle和BEA工作，任职华东区售前部门经理。2004年开始为企业客户提供解决方案咨询服务。2007年至今一直专注于数据库增值方案的业务拓展。

■ 责任编辑：李雨来（liyl@csdn.net）

Business Intelligence，还有很长的路要走

——记SAP 鲁百年博士专访

■ 记者 / 李雨来

SAP-BO中国首席顾问鲁百年博士，有着十多年BI相关领域的经验

说起Business Intelligence，很多人会想到OLAP，或是沃尔玛的“啤酒和尿布”的经典案例。但Business Intelligence这个名词的起源却很少有人了解。在十多年前，业界流传着DSS（Decision Support System）这个概念也就是决策支持系统。随着时间的推移，到了20世纪末，有一位叫伯纳德•利奥托德的人首先提出了Business Intelligence这个名词。之后，BI这个名词就被人们广泛流传了。而伯纳德•利奥托德就是Business Objects的创始人。到了2007年，SAP完成了对Business Objects的收购使之成为了SAP的子公司。为了帮助大家揭开围绕着BI的诸多疑团，我们采访了SAP-BO中国首席顾问的鲁百年博士。

可能有很多读者要问了，BI的具体概念是什么？鲁百年博士讲道：“根据Gartner的定义，BI就是一些帮助企业做分析决策的工具。这些工具，应该包含以下几个方面的功能：第一个是报表，第二个是随机查询，就是OLAP查寻，第三个就是预测，最后就是分析。”

处于断层期的BI

对于Business Intelligence这个概念，已经提了很多年了。但是BI这个技术现在到底处在什么阶段呢？鲁百年博士认为，目前的BI还仅仅是个开始，它目前应该处在“断层期”。

为了解释BI所处的阶段，鲁百年博士画出了产品生命周期图如图1所示，并解释到：

“这张图描述了一个产品从研究阶段到生命周期终结其成熟度和客户接受度的变化。任何一个产品在研发之前就已经开始做研究，之后会有一些第一个吃“螃蟹”的用户，这时其产品的客户接受度逐渐提高。但由于新产品不成熟，这时一些先行者有可能就成为了“先烈”，很多企业维持不了，在这地方死掉了，这个时候叫断层期。随后当所有人都认为这个东西赚钱了，已经快到成熟阶段的时候，就是中间成熟期，之后就是平稳期。”

鲁百年博士认为BI这个产品其所处的位置刚好在“断层期”这个区间当中（也就是两条虚线当中的某一位置）：“为什么要说在这个前后呢？什么时候到普及期？人人都需要把它当成桌面系统的时候，那是普及期。人人都需要，我认为现在没有到这个地步，既然没有到这个地步，那它肯定在成熟期之前。因为现在很多企业还把BI认为是锦上添花，不是说我是业务系统非要不可。所以说应该在断层期阶段。”

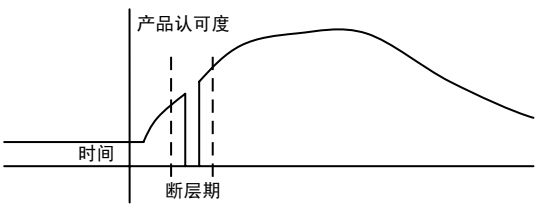


图1 一个产品的生命周期

谈到在BI这个领域的有实力的企业时，鲁百年博士提到了5家非常有实力的企业：SAP、Oracle、IBM、SAS和微软，并一一解析了这5家企业的现状：

因为SAP收购了BO，SAP的BI实际上也就是BO加上他原来的那个BIAR，还有他自己最近的数据仓库的工具BW。

对于Oracle来说，它旗下有3个BI的产品。首先是Oracle自己的BI系统；之后Oracle有收购了Hyperion，而Hyperion的BI是收购了Brio之后才有的；最后，Oracle又收购了Siebel的CRM，而这个CRM中也包含了BI。

IBM是在2007年11月收购了Cognos，Congnos有一个多维的数据存储，这个多维数据存储加上IBM原有的系统就构成了IBM的BI。

SAS，他是一家上市公司，但是他不应该叫BI，因为从国际BI的定义上来讲，我认为他不完全是BI，他应该叫数据挖掘，Data Mining，叫决策支持。

谈到微软，鲁百年博

士的语气跟介绍前面4家公司时不太一样："微软是我认为在商务智能领域里边比较让人头疼的公司，也是比较可怕的公司，虽然现在大家不太认可，但是他们现在跑的很快，我一直把它的产品叫贫民化，便宜、实惠、简单，人人都能用。"

BI遇到的问题

对于BI来说，从其诞生到现在，遇到的问题还是比较多的。很多实施BI的企业发现BI对自己的企业并没有太多的帮助，到最后，花了钱而不了了之。那么我们的BI到底遇到了哪些问题呢？在于鲁百年博士的谈话中，他提到了下面几个问题。

对BI的目标用户理解有问题

对于这个问题，鲁百年博士讲道："人人都认为商务智能就是决策支持系统，一说到决策支持谁用呢？老板。一说到老板，这套系统做完了以后就是给中高层的管理者用的。那么有个问题就出来了，一个公司的中高层管理者有多少人？有多少人在用BI？那些人会不会电脑？如果他们不会，也不会用BI，那么BI纯粹就是一个摆设，这套系统就该牺牲掉了。以前的商务智能之所以没有做好，都是因为认为BI是给老总用的，花了那么多钱，结果谁也不用，一旦不用了，那这套系统用来干吗？"

BI系统本身的复杂性问题

在谈到系统的复杂性问题时，鲁百年博士打了一个傻瓜相机的比喻："现在看来每一个企业，每一个地方都需要BI。既然这么需要BI，那为什么不能推到每一个人面前呢？原因很简单：BI工具太复杂。举个照相机的例子，照相机原来个头很大，携带起来也很困难，没有专业人士，谁能照相？现在照相机呢，因为它傻瓜、好用、简单，人人手上都有照相机，那么商务智能也一样，之所以走不到每个人的面前，跟它的复杂有很大关系。"

企业自身的信息化建设不够

在谈到企业信息化建设方面，鲁百年博士讲到一个企业的信息化系统建设应该分三个层次：核心业务系统、管理信息系统和商务智能系统。

核心业务系统就是承载企业运营的

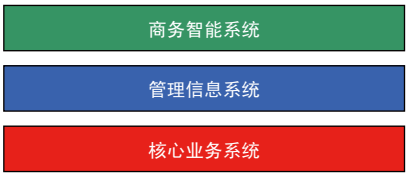


图2 企业的信息化系统建设的三个层次

系统，它应该保证实时准确。在核心业务系统之上是管理信息系统，不过这个"管理信息系统"不同于MIS。管理信息系统中包括了对企业上游（SCM）、中游（ERP）和下游（CRM）系统和对其的整合。与之相称的我们也可以通过最近比较火的一个名词CPM（全面企业绩效管理）来理解管理信息系统。最上层的就是商务智能系统。而商务智能系统最重要的是要有数据来支持其分析。

那么信息化建设的问题在哪里呢？对此鲁百年博士强调了："商务智能为什么不能走到贫民化，最主要的原因是，很多企业的业务数据没有，信息没有，因此并不是每一个企业都能马上去上BI的项目。"

让BI走下"神坛"

前面，我们已经了解到了这么多在BI实施中的问题，那么这些问题到底应该怎么解决呢？鲁百年博士给我们开了几个药方。

对于第一个问题来说，鲁百年博士认为BI系统应该是给每个人用的系统。

对于公司不同层面的用户，鲁百年博士介绍了一个清晰地分层的结构：对于老板和高层管理者来说，BI提供的是对企业运行状态的一个检测结果，体现到BI工具上就是仪表板；对于企业的中层管理者来说，BI应该提供例外分析功能，让这些管理者能清楚地了解他所管辖的部门在运行中哪些地方出现了问题，从而去寻找应对方法；而对于企业一线的运营层面的人员来说，BI应该提供一些详细的报表，以指导他们更好地完成日常的工作。如果这3个层次的人员都能用好相应的BI工具，那么BI的实施就达到了一个很好的效果。

对于BI工具的复杂性问题，鲁百年博士认为，解决这一点是许多BI厂商在努力的目标，而且已经有了一些进展了。

对于第三个问题，答案显而易见，就是企业要有自己的业务数据。不过这说起来容易做起来难，而且不同的企业所面对的情况也不一样。

更容易使用的BI工具

谈到如何让BI的工具更易用，鲁百年博士很兴奋地介绍了SAP Business Object的水晶易表。对于BI工具来说，鲁百年博士一直强调其易用性，而水晶易表则完美地阐释了这个思想。

对于BI工具的未来，鲁百年博士表示它应该更接近于Word、Excel、PowerPoint这些日常的办公软件，并与它们整合。这样我们可以更容易地在日常地工作当中使用BI工具，为我们的决策提供支持。随后鲁百年博士为我们展示了一下水晶易表的功能。它可以与Word、PowerPoint、Excel整合，通过简单的操作，就可以在PPT中加入BI工具的分析功能。比如说在PPT中的加入企业运行时的仪表板，或者在PPT中加入企业成本分布图表，而且还可以对这些图表的参数进行修改，然后直观地看到参数改变后的结果等等。

后记

对于BI我们还有很多认识是错误的，要改变这些错误的认识需要时间；对于BI工具来说，它还可以改进地更加简单易用；对于BI的使用，我们还需要让其更多地服务于企业的一线员工，让更多的人来享受到BI带来的好处。

面对BI，我们还有很长的路要走。■

■ 责任编辑：李雨来（liyl@csdn.net）

与国际化的信息化产品正面竞争

——黄涛谈用友U9的5年研发路

■ 记者 / 欧阳璟

“今天的企业管理软件，尤其是大型企业管理软件，市场几乎已经被SAP、Oracle这样的软件巨无霸瓜分得所剩无几。”这是5年前用友软件面临的情况。那个时期，用友的U8产品在市场上还是具备一定的竞争力的，然而其主要业务都是面向中低端企业用户。而面向高端的NC产品在那期间还显得不够成熟，并且主要关注在集团企业财务的业务问题上。在高端领域，国内软件产品几乎没有任何份额。在这种复杂的环境下，要决定开发一款中高端的企业级ERP产品，是需要很大勇气的。

竞争产品解决什么问题？

当时为了这个目的从金算盘离职，并进入用友、现在担任用友软件股份有限公司副总裁兼U9研发本部总经理的黄涛，对这个问题其实有他自己的看法：“面临如此强大的竞争对手，用友自己的产品竞争力在哪里？这个问题是我们在当初做U9的时候，要回答第一个问题。我们为什么要做U9？ U9为企业解决什么样的问题？做U9采用什么方式才能成功？在产品研发之初，我们花了很长时间来反复进行讨论和研究。当时我们对市场的分析结果显示：用友在当时国内的主要企业软件产品中，中低端应用软件是比较有优势的，尤其是在财务软件领域。而在中高端，特别是集团财务领域，用友的产品也有其一定的优势。但是在企业信息化这个巨大的市场面前，也就是通常认为的中高端ERP、中高端财务供应链制造、综合一体化，国内的管理软件的核心竞争优势几乎是空白。正因为这种空白，我们才要去填补它。”

“那时用友产品研发的主要手段是对现有产品的升级和改造。由于用友原来的产品定位，主要是从财务管理软件逐步延伸出来的，因此产品在架构上以功能驱动比较多，流程驱动比较少；以财务为核心比较多，以业务为核心比较少。当然，这种说法仅仅只是从当时用友现有产品的功能角度而言的。从技术上来看，我们则觉得产品本身的技术架构老化，需要一个新的产品来全面解决这些历史问题，而不再是修修补补地小打小闹。”黄涛在谈到最初准备开发新产品的时候仍然记忆犹新。

图1 用友的产品结构

当然，对于用友来说，决不能仅仅凭感觉来开发一个这样巨大的产品。这种规模的投入可能意味着公司将倾尽全力来打造一个旗舰产品，这也预示着这个产品的开发只许成功，不许失败。而最关键的角色：用户，提出的建议更值得用友思考：“当时我们聆听了大量客户的使用反馈。多数客户认为，在跨越式成长的中国环境下，很多商业模式都是完全创新的模式，即使购买SAP、Oracle的产品，也未必就能满足这些创新的思路。结果不出所料，很多用户的使用情况并不理想，新商业环境、新业务、新流程，就算是国外软件也无法做到。更困难的是，国外软件产品很难体会国内的需求，某些业务的使用方法，比如分销的模式，国内和国外在业务本身上的差异很大。而在这个领域快速成长的国内企业对应用的需求不断升级，因此需要新一代的软件。”

除此之外，黄涛还考虑了当时国内用户对软件和管理投资的看法：“国外厂商提供产品和服务价格是比较高的。然而这些产品与服务对企业来说性价比却未必很高。除了那些规模巨大的企业，

通常拥有很高的管理和IT投入预算，才能承受这么高昂的费用。而对于那些在数量上占很大比重的优秀中层企业，花大价钱投入到管理软件上还是有不小压力的。这些企业每年几亿、几十个亿的收入，如果让他每年拿出几十万来支持IT还能应付，但要掏出更多的钱来做这件事就非常困难了。这也给了我们这样一个机会。”

“而从地域的角度看，欧洲有一个SAP是德国企业，美洲有一个Oracle是美国企业，在亚洲能不能有一个用友是中国企业呢？”黄涛补充到。

规划后两件重要的事

前所未有地被寄予希望，U9开始了漫长了产品规划之路。“整个U9的规划期花了半年多时间”黄涛介绍道：“这期间我们做了很多工作。比如说预算问题，我们设想的这个产品需要花费1000个人/年的时间进行投入，量化到投入资本上看，就是4亿。总体的预算并非凭空设想而来的，这当中包括了整个产品的设计方案，包括技术方案的确定、产品商业模式、软件业务规划、产品与服务设计、架构设计、关键应用规划等多方面的内容，最终得出了这个数字。期间，我们还从公司内部的市场、技术、产品、服务等部门借调了不少力量参与到项目规划当中，并成立了产品规划小组。此外，我们还邀请了大量的外部专家，一线客户等外部资源共同协调，组织了很多次研讨会。另外，公司还花钱购买了一些国外的分析报告等。”

为了要和国外的产品竞争，国外的对手情况是必须要了解的。然而在完成了整个产品项目初期的规划后，整个技术领域的变革到来了。这也让黄涛对整个产品的计划进行了一定的调整。

“最初的计划并没有5年时间这么长。如果就开发一个纯粹的产品而言，两三年时间已经足够了，在整个U9的研发过程中，我们做了两件很重要的事。”

黄涛所指的两件很重要的事是指平台与业务模型。这是产品的设计思路与技术上的工作。第一件事是平台。

“从产品角度来看，第一件事是要做UAP平台。这并不是通常意义上所说的‘磨刀不误砍柴工’，而是真正意义上的整个用友基础设施的平台。事实上，U9的开发是先将UAP平台开发出来，再在这个基础上开发U9。从架构层面看，用友应该有一个统一的架构来适应企业应用开发，这是我们第一个目标。第二个目标是实现企业应用架构。平台层应该有领域的描述，企业应用上建模的方式。真正的具体的服务，是每个行业、每个地区，甚至每个客户都会不一样。如果要用平台层将这些都整合在一起，无疑是件很可怕的事。所以平台本身就有分层架构，层层分隔开后，最下面一层肯定是跨平台的，而上层可能有些服务适合某些领域，它是由宽到窄的模型。”黄涛一边说，一边用手比划。

第二件是业务模型。由于U9的开发过程中遇到了企业级软件的一个重大变革，那便是SOA，而业界讨论SOA也不仅仅是在技术层面上，而是从企业重新构建业务模型角度来考虑。因此U9在产品开发过程中参考了大量SOA的模型。“在U9开发过程中，我们定义了一个企业运作组件化模型，用组件描述服务。然后，SOA架构跟IT的组件对应，这样才能实现SOA架构对IT系统的承诺，从而达成业务的敏捷性。这个SOA的模型结合了用友在企业级领域20年的经验，包括客户的积累以及新管理模式的发展。这项工作花费了整个研发过程两年的时间。也正是因为这两年的时间，确保了我们的产品在面对竞争时能保持其足够的先进性。”

技术决策并不难

大多数企业级产品的开发都涉及一个重要的抉择，那就是技术架构的选型，.NET还是Java一直是开发者社区喋喋不休的争论话题。但在黄涛眼里，这个问题并不困难：“当时针对技术选型问题，公司内部意见并不一致，争论也很大。但到了最后，这个问题其实很简单，因为选择权在客户而不在厂商。大中型的客户在客户端上几乎全部采用了微软的平台，因此我们也就理所应当地选择了微软技术。”

然而，内部的反对观点仍然不能忽视：“当时主要持反对意见的人，一个重要的观点在于跨操作系统。实际上操作系统的跨平台对企业应用来说并没有太大意义。可以说我们的用户对跨平台特性的关注并不如我们想象的大。他们关注的是其他的问题：能不能实现高性能、大数据量的交互等等。这些特性是用户非常关心的，它们只对结果感兴趣，而不是手段。在系统平台的选择上，因为大中型企业的IT的现状如此，甚至整个业界的操作系统平台使用情况都是如此，所以选择的余地并不大——几乎都是微软平台。用户觉得采用微软技术的总体拥有成本更有优势，所以我们只好选择微软平台。可以说，选择什么样的技术平台，已经不再是产品竞争力的核心优势之一了。”

对于U9产品本身而言，难点主要还是在其业务模式的创新上。经过5年时间，数百人持续不断的努力，2009年，U9产品已经达成批量交付的目标。回顾2006、2007年两年的时间，黄涛说：“真正花在U9产品开发上的时间，持续了两年多。最后我们统计了工程量，整个产品的代码数量超过1000万行。”即使在国外，这种规模的代码量也绝不多见，更何况这还是一个批量交付的产品。U9的开发完成，可以说是国产软件与国际化企业软件产品的一次正面竞争，黄涛表示：“随着技术的发展，我们抓住了后发的优势，新时代的到来也将给用友走向国际舞台的机会。”■

■ 责任编辑：李雨来（liyl@csdn.net）

新产品 & 工具

Office 2007 SP2

Office 2007 SP2在4月份通过Windows Server Update Services发布。SP2拥有很多重要功能，包括：支持保存为ODF和PDF格式；提升了Outlook性能，增强了Outlook日历的可靠性；修复了核心Office应用软件中的图表漏洞；提供了SP客户端卸载工具；新增了很多用户对Office Server产品提出的改进；Office 2007 SP2包括自Office 2007以来的所有补丁。

GreenSQL 1.0

数据库防火墙程序GreenSQL，用来防止SQL注入攻击，其已经升级至1.0版本。最新发布的GreenSQL 1.0版本，在稳定性、易用性和性能上面均做了针对性的提高。包括代码优化、无标准的MySQL SQL 命令的拓展支持，网络连接相关的bug修复等。

GUIDE 1.0.0

GUIDE（GAIT Universal IDE）是由北航GAIT研究组开发的、专门为NOI选手设计的轻型集成开发环境。GUIDE具有跨平台、操作简单、支持C/C++/Pascal三种语言和单文件编译调试等优点。经过近一年的试用和修改之后，GUIDE 1.0.0版目前正式发布。 GUIDE 1.0.0版在测试版的基础上，根据用户的操作习惯对若干功能进行了调整，并添加了多个提升选手操作速度的快捷功能。本版GUIDE将为选手在练习和比赛中提供更多的帮助。

easyMule 2.0 For Linux

easyMule2.0是VeryCD完全重写的一个新版跨平台（Linux，Win，Mac）电驴骡，目前还处于测试阶段，虽然还不少bug和稳定性方面的缺陷，却是一个值得期待的产品。各类流行软件向Linux平台的迁移，是广大Linux爱好者和用户乐意看到的，也是Linux平台逐渐普及的一个象征。

ASDL 1.4.0

ASDL全称为Application Server Development Library，即“应用服务器开发库”，是一个用于在多个平台上进行应用服务器、网络程序及普通程序的通用组件库和开发框架。项目的创立初衷是提供一个高效的、可复用的、易用的组件库及框架，供快速开发网络程序及普通应用使用。

ASDL 最新的1.4.0 版于近日发布，其所对应的泛用型数据结构模版库（GDST, Generic Data Structures Templates）为1.0.1。该版本中为所有平台提供了基于TCP/UDP的同步、异步服务器，以及基于UDP的iso、epoll、kqueue服务器，修正了部分bug和接口。

SQL Server 2008 SP1

4月8日微软正式发布了SQL Server 2008的首个升级服务包，所有版本均可下载SP1进行升级，SQL Server 2008 SP1支持Windows Server 2003、Windows Server 2008和Windows Vista。SQL Server 2008 SP1并没有重大更新只是对其配置能力进行了提升，精简了安装配置步骤，对很多方面进行了完善，主要为以下三点：Slipstream将允许用户同时进行对SQL Server 2008和SP1的安装，减少整体安装时间；新增了SP卸载功能；Report Builder 2.0新增了对Click Once的支持。

Andriod 1.5 SDK Preview

Google发布了Android 1.5 SDK，开发者能通过它来预览到Android平台的下一代版本。Android 1.5基于Android项目开源分支Cupcake，不但支持虚拟键盘操作还增加了视频录制功能，支持蓝牙立体声、配备加速器，加快对屏幕倾斜旋转的控制速度。同时在Web浏览器上，同样增加了复制、粘贴文本功能等等。新引入的API特性包括软键盘、主屏幕Widget、Live文件夹及语音识别。其它的特性包括 UI改进，性能增强——更快的照相机启动速度和照片拍摄，更快地的读取进行GPS定位等。

Microsoft Desktop Optimization Pack 2009

微软企业桌面优化套件（Microsoft Desktop Optimization Pack，简称MDOP）是一项采用了创新技术的动态桌面解决方案。与Windows Vista的配合使用，可扩展Windows Vista的价值，提供更高优化程度的桌面，也就是实现用最节约的成本、最灵活的方式管理Windows桌面的目的。

新发布的MDOP2009中包含：Microsoft Enterprise Desktop Virtualization，Cumulative Update to Microsoft Application Virtualization，Updated Asset Inventory Service；三个标准 MDOP 工具：Microsoft Diagnostics and Recovery Toolset, Microsoft Advanced Group Policy Management, Microsoft System Center Desktop Error Monitoring。

Chomium Build 13360 for Mac

Chomium的Linux版本一直处于开发阶段，这次Google首先放出了基于Mac的测试版本。基本功能包括：基本网页浏览；书签页功能；新标签页中的最热门浏览；打开新标签页；返回、前进、重新加载；全屏浏览；新窗口中打开链接；拖动标签建立新窗口；运行新标签页；剪切、复制、粘贴；键盘快捷键；部分about页面，包括about:version、about:dns、about:crash、about:histograms。

QuickOffice for iPhone

移动办公软件Quickoffice发布iPhone版本，包含文件共享功能和协作功能，不同终端之间可以传输文件，并可以从桌面使用Wi-Fi远程访问其MobileMe的iDisk帐户。iPhone版的Quickoffice具备基本的剪切、复制、粘贴功能，除了可以编辑Word和Excel文档，还可以查看iWorks、PDF和其他常见的媒体文件，支持自动保存和自动恢复功能，避免丢失重要资料。

ImgBurn 2.4.4.0

ImgBurn是款支持多引擎的刻录软件，定位于轻量级刻录工具。支持所有主流刻录机和所有主流光存储媒介，此外还具有支持诸多先进的防刻死和刻坏盘的特性，可外挂调用包括Nero在内的多款商业刻录软件的光存储引擎，甚至还支持命令行操作，便于第三方软件调用ImgBurn。自2.2版开始新增了对UniCode的支持。除了微软的SPTI以外的第三方引擎进行刻录，ImgBurn还集成了若干其它引擎，包括Nero ASPI、ElbyCDIO和VSO Patin-couffin。

Windows Server 2008 Foundation

对于中小企业来说，Windows Server还是一个应用很广泛的服务器操作系统。就在4月，微软发布了Windows Server 2008 Foundation。这个新发布操作系统主要是针对中小型企业，只支持单路单CPU的硬件，而且不支持Server Core和虚拟化技术。不过这款操作系统只会以OEM的方式安装到新的服务器上，所以其优势是中小企业可以以更低的成本购得一台装有正版Windows Server 2008的服务器。

Microsoft Photosynth

Photosynth是一款以处理照片为中心的基于B/S模式的软件系统。通过图像要素提取分析算法，将大量相关的不同来源不同视角的二维照片处理成为拥有三维属性的立体景像。根据处理后的海量照片信息，能够及时的展示其他具有和你当前观看的图片相类似特性的图片，注释、标签或者URL链接等都能够和一个图像关联，并能够传送到相似的图像中去。该软件能自动分析某物体或者某个场景的一组任意角度、任意时刻、局部或整体的照片，智能辨识出每张照片之间的空间对应关系，从而生成可以自由旋转、察看的三维全景模型。目前最新版本是2.0109.0415.1554。

SpacePilot PRO 3D

3DConnexion SpacePilot PRO 3D鼠标是专门为处理三维图形而设计的专业鼠标。专业人士操控此鼠标，可以非常方便的实现三维模型的旋转移动等操控。另外这款鼠标还配备了一个LCD彩屏，以及很多快捷控制按键，还支持查看电子邮件、日历等功能。

TASCAM FireOne

TASCAM FireOne是TASCAM和FRONTIER一起设计的组合式便携火线音频接口。火线音频接口是TASCAM成熟的产品了，在此基础上加上FRONTIER控制器的经验，就结合出了这么一个看起来很怪异的FireOne火线音频接口+控制器。具有人体工程学设计的控制键和大号闪灯控制轮，控制功能强大，软件支持的很好，可自定义功能键功能。

Acer AspireRevo

Acer推出的基于nVIDIA ION平台的轻便省电的桌面电脑。采用1.6Ghz的Atom 230处理器，支持4GB内存、250GB硬盘、卡片阅读机、HDMI/VGA，大小为7.1×7.1×1.2时，支持7.1声道。

Coral

Coral 是BenQ—Siemens尚未生产的概念型手机，上滑为音乐模式，下滑为手机模式，造型在现在看起来也颇为时尚。

Lyeio

UWB（Ultra Wideband）是超宽带无线技术的缩写，Lyeio UWB是利用此技术的一种个人信息管理设备。内置16G Flash内存，1.5寸 OLED 屏幕，具备三轴加速传感器及震动功能。Leyio之间可通过UWB技术交换数据。支持指纹识别，只有经允许的使用者才能够读取交换数据。

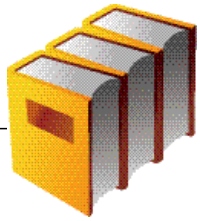
D-Link DHP 电力猫

D-Link新款电力猫将可以充分利用普通电话的电线以实现高速网络接入。其使用的是方便的即插即用设计，共有两个插头/适配器，非常适合于基于网络的数码播放器，也可以用来与游戏机配合使用。

I-Sobot

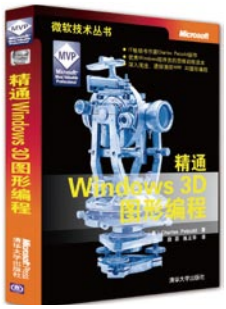
TOMY推出的世界最小的ISobot机器人。双脚可步行，只有16.5CM的机身。装有自主研发的超小型伺服器马达17个，装有微型陀螺盘感应器（陀螺仪），可进行多种顺畅动作。

新书上架



精通Windows 3D图形编程

作者：（美）Charles Petzold
译者：段菲 陈正华
出版社：清华大学出版社



书中内容围绕用WPF 3D API进行3D图形编程而展开。针对3D的基本要素及其在WPF中的工作原理，作者通过丰富的图片和大量XAML/C#代码，进行详细的解释和适度的探究。全书共分9章，内容涉及网格、模型、摄像机、光照、材质、变换和一些基本的3D数学知识，书中素材经过作者精雕细琢，具有丰富的实例，必要的地方还有详细解释。

阅读此书，读者将掌握如何利用Microsoft .NET Framework 3.0和Windows Vista进行3D图形的显示和动画处理。书中包含的专家指导和XAML/C#实例，有助于读者掌握适当的技能，创建出高度逼真的用户界面。作为一本了解3D图形编程的指南之作，该书适合于具有WPF基础或熟悉其他3D平台（DirecX/OpenGL）并希望深入了解WPF 3D的读者阅读。

“面向对象”项目开发经验大成：基于.NET实现

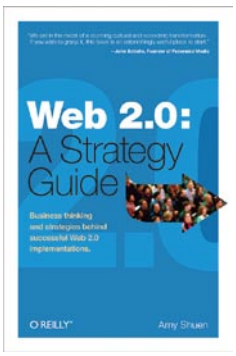
作者：牛树长 杨海波 俞丹琛 黄智洪
出版社：电子工业出版社

此书基于开发过程的各个环节，总结筛选了一批可以被“复用”的命题素材，并将其创造成极具代表性的设计样例，以此由浅入深地诠释面向对象的设计理念是如何被应用于实践的。书中基于.NET C# 实现的具体样例，并不局限于某个项目的具体应用，而是侧重于通用、共性的特点，当把这些通用、共性的设计成分组织成一个相对完整的体系之后，就形成了一个具有“复用”价值的开发工作环境。以这些通用设计为基础进行应用项目的开发，可以大大提高开发效率、降低对人员素质的要求。

作者认为，只要能坚持面向对象的设计理念，做到自定义“构件”一点也不难。在你设计了一批“构件”并能形成体系的时候，自然就会发现“构件”化的设计体系也不过如此。同时也会心生感慨：面向对象的殿堂原来是如此壮观与辉煌。

Web2.0策略指南

作者：（美）艾米
译者：赵俐 盛海艳 等
出版社：机械工业出版社



Web 2.0已成为新闻焦点，但它如何赚钱？作为一个简明指南，《Web2.0策略指南》解释了Web 2.0有什么特异之处，以及这些特异之处如何帮助公司赢利。

书中示例关注的是Web 2.0的效率，而非聚焦于技术。你将了解到这样一个事实：创建Web 2.0业务或将Web 2.0战略整合到现在业务中，意味着创建一个吸引人们前来访问的在线站点，让人们愿意到这里来共享他们的思想、见闻和行动。当人们通过Web走到一起时，结果可能远远大于各部分的和。

无论你是一位正在谋划下一步行动的执行官，还是正在寻找扩张途径的企业主，或是正在规划下一个创新业务的企业家，该书提供的真实生活中的示例展示了各种规模的公司如何在当今的网络上创造新机遇，相信定会为你打开另一扇窗口。

数据重现
——文件系统原理精解与数据恢复最佳实践

作者：马林
出版社：清华大学出版社

该书是国内第一本全面介绍Windows及非Windows文件系统的数据恢复技术书籍，涵盖面广，内容深，为畅销书《大话存储》的姊妹篇。书中不仅对常见的DOS分区体系及Windows的FAT文件系统、NTFS文件系统进行了详细介绍，更涵盖了苹果机分区、BSD分区、SPRC平台的Sun Solaris分区、GPT分区等分区方式，以及对Linux的Ext2/Ext3、Unix的UFS1/UFS2、MAC的HFS+等文件系统布局及详细数据结构的讲解，多数资料的详细程度是目前少有的。作者同时还对常见RAID类型及包括HP内外双循环、RAID 1E、RAID6及RAID DP在内的异种或新型RAID类型进行了详细的分析和介绍。

此外，作者从数据恢复前的准备到实际恢复工作的进行，一步步带领读者踏入数据恢复的殿堂。

数据仓库结构设计与实施：建造信息系统的金字塔（第2版）

作者：池太崴

出版社：电子工业出版社

今天，越来越多的部门和机构开始接受并开发数据仓库，并把它作为信息集成的解决方案和决策支持系统工具，以迎接日趋激烈的社会和商业管理的挑战和竞争。

本书从数据仓库技术背景、技术结构框架、开发和应用等方面，结合作者在数据仓库技术实施过程中的实际经验，深刻阐述了数据仓库开发生命周期在各个阶段的特点和策略运用，以及在管理信息系统中的位置和作用。

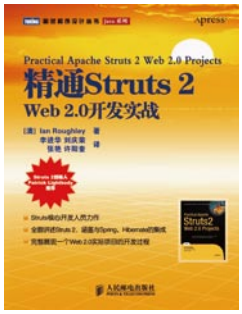
无论我们希望打造一个什么样的信息系统，结构总是一个需要认真考虑的首要问题，本书旨在介绍和探索数据仓库的基础技术和结构概念（如多层次结构），因为结构设计为数据仓库开发以及各种决策支持系统奠定基础。当我们对众多信息系统开始进行以集成为目标的基础结构改造的时候，这些将变得更为重要。

精通 Struts 2：Web 2.0 开发实战

作者：Ian Roughley

译：李进华 刘庆荣 张艳 许阳奎

出版社：人民邮电出版社



Struts灵活易用、功能强大，是广受欢迎的Java Web框架。新版本Struts 2更上一层楼，提供了更好的AJAX和Spring集成支持。书中通过一个完整的Web应用示例，全面讲述了Struts 2框架本身以及运用Struts开发Web项目的全过程。作者通过这个循序渐进的开发应用，全面直观地展示了如何运用Struts 2的各种特性，结合Spring、Hibernate和各种Web 2.0技术，创建功能强大、流畅易用的网站。在此开发过程中，读者还将对Struts框架、Web开发和现代软件开发实践有更加深入地认识，从而提升自己的编程功力。 此书作者Ial Roughley是著名的Java技术专家、Apache Struts项目管理委员会成员、From Down & Around公司创始人，著名技术网站IntoQ Java领域的编辑。他在软件架构、开发和过程改进方面都有丰富的经验。

思维导图

作者：（美）东尼•博赞

出版社：中信出版社

《思维导图》是一本在全球销量达百余万册的畅销书。书中的思维导图方法将放射性思维和开拓性笔记技巧结合在一起，被人们称为“大脑瑞士军刀”。它的出现，在全球教育界和商界刮起了一场风暴，目前全世界已有超过2.5亿以上的人在使用。

本书讲解了著名的“思维导图”，逐一教授读者形成一套属于自己的、简单的、能够帮助组织和自身迅速提高绩效的思维、管理和学习工具！它可以让读者重新认识大脑，学会迅速剖析复杂问题、把握全局，有条不紊地推进项目的能力，并能快速收集和处理大量信息，焕发读者创造灵感，享受深入思考、分析问题、解决问题的乐趣，帮助读者成为高效人士。书中内容，可增强使用者的记忆能力、立体思维能力和总体规划能力，并逐步提升学习和阅读能力，进一步激发读者的创造性和想象力。书中还特别提供了一系列刺激练习和颇具启发性的图片集和演示这种技巧的原创性思维导图。

作者东尼•博赞是“思维导图”的创始人，长期从事大脑的思维研究工作，被业内人尊称为“世界大脑先生”，还被誉为英国的“记忆力之父”，是大脑和学习方面的世界顶尖演讲家。

全球图书销售排行榜

	Amazon	第二书店	天龙书局（台湾）
1	Beginning iPhone Development: Exploring the iPhone SDK	卓有成效的程序员	大话设计模式
2	The Black Swan: The Impact of the Highly Improbable	iPhone 开发基础教程	ASP.NET 专题实务 - 使用 C#
3	The Hot Shoe Diaries: Big Light from Small Flashes	云计算	Linux Device Driver Programming 驱动程式设计
4	Pokémon Platinum: Prima Official Game Guide	走出软件作坊（三五个人十来条枪如何成为开发正规军）	jQuery 实战手冊
5	Presentation Zen: Simple Ideas on Presentation Design and Delivery	项目管理之美	Silverlight 2.0 精华技术手册—使用 VC# + WPF 程式设计
6	Programming in Objective-C 2.0 (2nd Edition)	编程之美——微软技术面试心得	个人稳私泄出与防护手法大公开
7	The New Rules of Marketing and PR: How to Use News Releases, Blogs, Podcasting, Viral Marketing and Online Media to Reach Buyers Directly	PHP 和 MySQL Web 开发	资讯安全基础概论
8	Presentation Zen	庖丁解牛：纵向切入 ASP.NET 3.5 控件和组件开发技术	Google！ Android 手机应用程式设计入门
9	Mac OS X Leopard: The Missing Manual	jQuery 实战	铁的机密档案捍卫法则
10	World of Warcraft: Arthas: Rise of the Lich King	深入理解计算机系统（修订版）	Silverlight 2.0 范例权威讲座

蒋清野，1999年获得清华大学学士学位，2000年获得美国伊利诺大学（University of Illinois at Urbana-Champaign）硕士学位，现任Sun中国技术社区高级经理。

从Oracle收购Sun公司谈起

■ 文 / 蒋清野

4月20日，Sun公司董事会通过决议，同意以每股9.5美元的价格将公司出售给Oracle。一时间，很多IT领域的朋友众说纷纭，对Sun公司各种产品和技术的前途提出种种预测。在这里我个人就Sun公司软件部门的一些产品和技术的未来发表一点看法。需要说明的是，虽然我个人在Sun公司工作，但是并没有任何机会接触到公司决策层的任何相关资料。因此，在本文中出现的所有观点，仅仅是我个人的观点，而不是我的雇主Sun公司的观点。

Solaris操作系统

在Oracle与Sun公司共同发表的新闻稿当中指出：Sun公司所拥有的Java语言和Solaris操作系统是“根本性的长期战略优势”。Solaris操作系统对于Oracle的重要性，由此可见一斑。

作为一家数据库厂商，Oracle做梦都想要拥有自己的操作系统。在没有自有操作系统的情况下，唯一的选择就是全面支持市面上的各种操作系统，根据市场的变化来调整不同操作系统的优先级。2000年之前，Sun公司正如日中天，在金融、电信、能源等多个关键性领域的装机量均排名第一，因此Oracle选择将Solaris作为优先考虑的操作系统。2001年前后，GNU/Linux在服务器端的性能已经相当出色，可以运行在价格低廉的x86处理器上，能够方便地从网络上免费下载到安装文件。更重要的是，在网络上很容易找到与GNU/Linux相关的各种文档。与此相反，Solaris需要运行在昂贵的UltraSparc处理器上（当时x86版本的Solaris 8已经可以免费下载，但是还远远没有达到健壮实用的程度），安装过程烦琐复杂，系统管理员还需要经过Sun公司的专门培训。随着互联网泡沫的全面破灭，企业对信息系统的性价比提出了越来越高的要求。在这种情况下，Solaris的新增装机量开始下降，无须财务主管审批即可立即部署的GNU/Linux开始占领数据中心。Oracle敏捷地注意到了这个趋势，于2002年推出了名为“坚不可摧的Linux”（Unbreakable Linux）的客户支持计划，开始向GNU/Linux倾斜。与此同时，Oracle开始加大在操作系统方面的投入，慷慨解囊资助GNU/Linux社区中的多个关键性项目，同时试图构造一个全新的GNU/Linux发行版。由于操作系统的复杂性，Oracle在自有操作系统方面的进展缓慢。由于与IBM公司的DB2存在正面竞争，尽管Solaris的市场正在萎缩，Oracle依然将Solaris作为优先考虑的操作系统。2006年，GNU/Linux在数据中心的新增装机量已经超过了50%，Solaris的新增装机量则降低到15%以下。这时候Oracle的首席执行长官Larry Ellison做了一个聪明绝顶的决定：将Red Hat的图标换成Oracle的图标，将“Red Hat Enterprise Linux”几个单词修改成”Oracle Enterprise Linux”，一举推出了与Red Hat Enterprise Linux完全兼容的Oracle Enterprise Linux，同时推出价格仅为Red Hat一半的客户支持计划。尽管业界对此众说纷纭，Oracle终究是依靠GNU/Linux社区的强大实力摆脱了对Solaris的依赖，同时也过了一把“自有操作系统”的瘾。

这也许是Oracle收购Sun的原因之一：Sun懂得操作系统。在文件系统方面，ZFS是目前为止功能最强大的文件系统；在应用开发方面，DTrace能够轻易从内核层和用户层寻找应用程序的瓶颈；在系统安全方面，Solaris Trusted Extension获得了最为全面的EAL 4+认证（包括LSPP,CAPP,RBAC）；在超级计算方面，目前世界排名第六的德克萨斯超级计算中心（Texas Advanced Computing Center）运行的是Solaris操作系统。（需要说明的是，排名前五的超级计算机运行的都是不同版本的

GNU/Linux操作系统。）尽管新增装机量的增长缓慢，但是在世界各地的数据中心里，依然有10%左右的服务器在运行不同版本的Solaris操作系统。通过收购Sun公司，Oracle可一举获得操作系统领域的核心技术、人才、声望、以及现有的客户。

因此，Oracle不会放弃Solaris操作系统。剩下来的问题，是如何继续Solaris操作系统的开发。Oracle是会继续支持目前的OpenSolaris项目呢，还是会采用传统的闭源方式？我个人的看法是，Oracle有可能在GPL授权协议（有可能是GPLv3）的框架下重新发布OpenSolaris项目。OpenSolaris目前最大的问题是硬件兼容性的问题。在x86/x64平台上，尚有大量的声卡、网卡、显卡以及其他外接设备没有OpenSolaris的驱动程序。类似的问题，GNU/Linux社区已经解决得比较好了。相关的驱动，基本上都是开放源代码的，只是由于GPL协议和CDDL协议之间互不兼容，使得OpenSolaris社区无法利用GNU/Linux社区的这些成果。的确，使用GPL授权协议使得GNU/Linux社区也可以充分利用OpenSolaris项目中诸如ZFS和DTrace等等亮点，但是从长远来看，恐怕是OpenSolaris项目得到的好处要更多一点。毕竟，对于一位普通的开发人员来说，没有ZFS和DTrace的GNU/Linux已经足够好用，但是没有声卡网卡驱动的OpenSolaris就不太好用了。

Java编程语言

谈起Java编程语言，我们首先要明确Java语言目前有三个分支：Java SE（标准版）、Java EE（企业版）和Java ME（嵌入版）。Java SE是Java EE和Java ME的基础，类似于国家自然科学资金的基础性研究项目，基本上是光花钱不挣钱的。Java EE可以认为是Java语言在企业级解决方案中的应用，大部分做Java的公司，除了Sun公司之外基本上都是依靠Java EE盈利的。Java ME可以认为是Java语言在嵌入式设备方面的应用。在2006年之前，厂商每生产一台支持Java ME的手机，就要给Sun公司支付一定的授权费用。而2006年11月，Sun公司启动了名为PhoneME的开放源代码项目，这个收入就变得不是十分可靠了。

Java EE

在如上三个分支中，Oracle最感兴趣的显然是Java EE。但是在收购Sun公司之前，Oracle已经收购了这个领域的大牛BEA。让Oracle暗自动心的不是Sun公司在这个领域的市场份额，而是Sun公司作为Java语言的发明者在这个领域的领导地位。可以想象，在两家公司合并之后，Oracle必然会将Sun公司目前的Java EE部门一分为二——写标准和申请专利的继续写标准和申请专利，开发应用服务器的则并入原来的WebLogic部门。至于Sun公司自己的应用服务器GlassFish，想来这个品牌是不会再用了。下一代的Java EE参考实现，叫做Oracle WebLogic显然要响亮得多。按照同样的推理，Sun公司原来叫做Java企业系统（Java Enterprise System, JES）的那套东西，本来占到的市场份额就很小，合并之后跟Oracle现有的中间件产品线发生冲突，所以也将逐渐退出历史舞台。

Java ME

Java ME是一个比较难办的问题。随着各种高速无线网络的普及，移动与嵌入领域正在变成下一个金光闪闪的企业级取款机，但是Oracle在这个领域却毫无经验。我个人的观点，是Oracle会让这个部门继续独立运作一段时间，但是会指派一位党委书记（有可能是从外部新招来的）前来参观学习。等这新来的党委书记熟悉了Java ME部门的业务之后，才开始对该部门进行调整，并且调整的幅度不会很大。

Java FX的去留，要取决于Java ME的命运。我们知道，不管Java FX的桌面版做的有多好，在桌面这个领域是肯定收不到支票的。Java FX如果想要挣钱，就必须能够在手机上流畅地跑起来——这个事情，不仅仅是改进Java FX本身那么简单，还需要考虑手机的处理能力，以及跑在手机上那个Java虚拟机的效率。

不管是Java EE还是Java ME，都严重依赖于Java SE。如果底层的虚拟机做的不够好，上层的框架搭得再好都是白搭。Sun公司自己的Java虚拟机，有很多独到之处，执行效率也不错。两家公司合并之后，Oracle之前从BEA那里获得的JRockit估计要遭殃。Oracle是一家注重实用的公司，因此Java虚拟机下一步将注重于提升服务器端的性能，某些只有桌面端才用得上得功能，其优先级估计就要低一点了。在这一点上，Java虚拟机和GNU/Linux近年来的发展趋势基本上是一致的。

> **Oracle对开放源代码的态度，可以用四个字来总结：拿来主义。**

NetBeans

熟悉Java开发的朋友可能会问：NetBeans呢？这可是Sun公司花了10年心血精心培养常来的宝贝。NetBeans是一个免费的产品，Sun公司大力发展NetBeans，指望的是开发人员通过使用NetBeans将其开发的产品部署在自家的应用服务器等部署环境上，这样才能够卖出产品和服务。在过去四年中，虽然NetBeans的市场份额逐年稳步提升，但是在拉动其他产品方面的作用非常有限。Sun公司之所以坚持做了下来，是因为过去十年的惯性实在是太大，要想停下来的话不管在内在外都免不了要大折腾特

折腾一场。换句话说，这些年来Sun公司大力发展NetBeans确实保住了面子，但是底下里却输掉了钱包。这种死要面子活受罪的事情，Oracle的Larry Ellison是断然不会做的。

MySQL数据库

2008年3月，Sun公司以10亿美元的代价收购MySQL，被Jonathan Schwartz认为是“现代软件史上最重要的并购案”。如此重要的角色，在Oracle与Sun公司共同发表的新闻稿当中却并没有提及。很多业内人士认为，由于MySQL与Oracle的数据库业务之间存在直接的竞争关系，很有可能会被Oracle所抛弃。如果我们将数据库业务这个市场看成一个整体，Oracle与MySQL之间毫无疑问是相互竞争的。但是，如果我们进一步对这个市场进行细分，结论就有可能不太一样。

原MySQL公司的首席执行长官Marten Mickos曾经说过：“如果要在开源软件上取得成功，那么你需要服务于：（1）愿意花费时间来省钱的人；和（2）愿意花钱来节约时间的人。”拿数据库这个市场来说，MySQL所服务的大部分是第一种用户，小部分是第二种用户；Oracle所服务的大部分是第二种用户，小部分是第一种用户。MySQL所拥有的用户数量更多，因为不愿意花钱的人总是比愿意花钱的人多；Oracle所拥有的用户质量更高，因为愿意花钱的都是优质客户。所以Oracle更多地被使用于中大型企业应用，而MySQL更多地被使用于中小型企业应用，不过这个界限并不严格。因此，MySQL的用户群和Oracle的用户群之间存在一定的重叠，但是重叠的程度并不是很高。

但是这并不代表Oracle对MySQL的用户不感兴趣。2005年10月，Oracle收购了与MySQL关系密切的InnoDB。当时InnoDB为MySQL提供一些事物和外键方面的技术，主要使用于比较复杂的应用当中，对于一般的应用基本上没有什么影响。Oracle收购InnoDB的本意是希望通过拿走MySQL中的优秀特性来打压MySQL，阻止MySQL进入中大型应用这个市场。遗憾的是开发人员普遍将Oracle的收购行为理解成对MySQL的恐惧，并且进一步得出MySQL的性能已经足以与Oracle相竞争的推论，反倒帮MySQL做了一次活生生的广告，加速了MySQL的普及。

现在一切都顺理成章了，通过对Sun公司的收购，不管你用的是MySQL还是Oracle，你都是Oracle的用户了。请记住，MySQL被Sun公司收购之后，基本上是处于独立运作的状态，其现金流还是正的。只要Oracle表示继续支持MySQL数据库，就能够赢得MySQL社区的支持，并在适当的时候向他们提供更好的（要掏钱的）产品或者是服务。在未来的两到三年里，MySQL还是会作为一个独立的产品存在。但从长远来看，同时维护多个具有相同或者相似功能的产品会造成用户的困惑。因此，MySQL最终还是需要融入Oracle现有的产品线，只是要等到Oracle将MySQL现有的用户群消化掉而已。

开放源代码

通过这笔金额高达74亿美元的交易，Oracle还将得到一个赠品：按照源代码的行数来计算，Oracle将成为世界上对开放源代码社区贡献最大的实体。从操作系统（OpenSolaris）到编程语言（OpenJDK），从数据库（MySQL）到应用服务器（GlassFish），从开发工具（NetBeans）到办公套件（OpenOffice），这些源代码几乎无所不包。这个赠品来得比较突然，估计Oracle还没有想好要怎样去处理它。

Oracle对开放源代码的态度，可以用四个字来总结：拿来主义。如果一个开源软件足够好用，直接拿过来集成到Oracle的产品中就是了，没有必要为其支付任何费用。用Larry Ellison自己的话来说：“我不能够给开放源代码软件开出上亿美元的支票，因为这并不能够使我们在竞争中得到优势。我们能够做的，别人也能够做。”可惜的是，Oracle收购了InnoDB，并没有拉拢到MySQL的用户；Oracle推出了Oracle Enterprise Linux，也并没有能够拉拢到Red Hat的用户。开放源代码的价值，在于围绕该技术所形成的社区，这个社区包括该技术的开发人员和用户。只有这个社区成了规模，才能够给社区领袖带来经济价值。在这一点上，Marten Mickos看得要比Larry Ellison更为透彻。

和Oracle相比，Sun公司在开放源代码方面显然拥有更多的实战经验。在过去的四年里，只要是可以开放源代码的软件资产，Sun公司基本上都开放源代码了——不仅仅是软件，连其最新版本的处理器UltraSparc T2的设计都是开放源代码的。问题在于，作为一家挂牌交易的上市公司，开放源代码的举措并没有能够使公司摆脱财务上的困境，最终被Oracle收入囊中。Sun公司的这些“实战经验”，到底有多少能够为Oracle所借鉴，是个值得进一步探讨的问题。

结语

我于2004年10月满怀仰慕之情加入Sun公司。在过去的4年多时间里，Sun公司一直处于动荡之中。每隔三五个月，就来一次结构调整，让大家紧张一阵。这几年来，大大小小的调整经历了不少，竟然慢慢地也就习惯了。这一次经济危机，心里也知道公司的情况非常紧张，但是没有想到竟然这么快就要被卖掉。就借用《金刚经》中的几句话，作为这篇文章的结语吧。

一切有为法，如梦幻泡影，如露亦如电，当做如是观。■

■ 责任编辑：李雨来（liyl@csdn.net）